MATROID

Mobile Application Transport Routing Optimized

Título: Aplicación para móviles de distribución de paquetería

Autor: Miguel Olmos Alabert

Fecha: junio 2013

Director: Luis Domingo Velasco Esteban

Departamento del director: Departament d'Arquitectura de Computadors

Titulación: Enginyeria Tècnica en Informàtica de Sistemes

Centro: Facultat d'Informàtica de Barcelona (FIB)

Universidad: Universitat Politècnica de Catalunya (UPC)

DATOS DEL PROYECTO

Título del proyecto: Aplicación para móviles de distribución de paquetería

Estudiante: Miguel Olmos Alabert

Titulación: Enginyeria Tècnica en Informàtica de Sistemes

Créditos: 22,5

Director: Luis Domingo Velasco

Departamento: Departament d'Arquitectura de Computadors

Centro: Facultat d'informàtica de Barcelona (FIB)

Universidad: Universitat Politècnica de Catalunya (UPC)

MIEMBROS DEL TRIBUNAL

Presidente: Nacho Navarro Mas

Vocal: Jordi Petit Silvestre

Secretario: Luis Domingo Velasco Esteban

CALIFICACIÓN

Calificación numérica:

Calificación descriptiva:

Fecha:

Índice de contenidos

Índice de figuras

1 Introducción

En este apartado se definirá el contexto sobre el cual se ha realizado el proyecto, la motivación a la hora de realizarlo y los objetivos que se pretenden alcanzar.

1.1 Contexto

En toda empresa de transporte de paquetería terrestre es muy importante realizar las rutas más eficientes, cortas y rápidas posibles, de forma que se optimicen recursos y se reduzcan los tiempos entre recogidas y entregas.

Realizar el cálculo de las rutas con dichas características no es una tarea fácil. De hecho, es homólogo al famoso problema del viajante (o problema del vendedor viajero), según el cual un viajante debe visitar N ciudades y ha de encontrar un camino que las recorra y vuelva al punto de inicio, pero sin pasar por una ciudad más de una vez. Dicho camino se denomina Camino Hamiltoniano y tiene un coste NP-Completo. Este camino hamiltoniano es el que se debería de encontrar en cualquier ruta de transporte para que esta sea óptima.

Otro punto importante y necesario para las empresas es la flexibilización de estas rutas una vez han sido definidas, de forma que permitan modificaciones en tiempo real y dando capacidad de reacción a los transportistas dependiendo de las circunstancias del momento y del entorno. Esto permite una reducción de costes en situaciones como, por ejemplo, un atasco, que en un principio no habría otra salida que esperar para llegar al destino con la pérdida de tiempo y dinero correspondiente, cuando quizás la mejor opción es cambiar el destino dejando el actual para más tarde.

También es necesario mencionar, en el mismo sentido de optimización, los aparatos grandes, pesados y caros que suelen llevar los transportistas. Sustituirlos por smartphones de gama media-baja, que hoy en día tienen potencial suficiente para ejecutar múltiples aplicaciones, sería un gran avance tanto en términos económicos como en versatilidad, ya que permite un cambio del dispositivo rápido y eficiente, por ejemplo a causa de una rotura que impidiera el funcionamiento de este.

Por último, cabe destacar la importancia de un panel de control mediante el cual se muestre la información necesaria de forma fácil, intuitiva y ágil; y permita introducir, modificar y/o eliminar los datos del sistema en los casos y condiciones necesarias.

1.2 Motivación

Situados en el contexto del proyecto, la motivación de este se basa en crear un pequeño sistema o prototipo que pueda resolver las problemáticas definidas en el punto anterior, de forma que pueda ser utilizado por pequeñas empresas que necesiten un sistema de gestión de la actividad principal y mediante el cual optimicen los recursos y les ayude a reducir costes.

A nivel personal, me llama la atención la programación de dispositivos móviles, concretamente con sistema operativo Android, y todo el potencial que se les puede sacar; gran parte de ello gracias a la multitud de sensores que contienen y que permiten recibir información sobre el entorno que les envuelve, como por ejemplo el GPS. Creo que los dispositivos móviles tienen infinidad de utilidades y funcionalidades a explotar y, concretamente a nivel profesional, un gran camino por recorrer mediante el cual pueden cubrir una multitud de necesidades de las empresas, que sin la existencia de estos dispositivos no sería posible cubrir o se basarían en sistemas que requieren un gran trabajo de implementación y un gran coste económico.

También me ha llamado la atención la forma en que se llevarán a cabo los cálculos, ya que estos al ser muy pesados y complejos, no serán realizados por los dispositivos móviles sino que se ejecutarán en un servidor central. Serán los primeros los que enviarán peticiones al segundo para realizar las diferentes acciones. Además te da una seguridad extra, ya que la rotura o pérdida del dispositivo no conlleva la pérdida de los datos.

Otra problemática interesante heredada al utilizar dispositivos móviles son las conexiones de estos al servidor y viceversa, ya que estas suelen ser débiles y con interferencias. Crear un sistema robusto sobre este problema, que puede generar una incoherencia en los datos del servidor central y provocar un error inmensurable, es un valor añadido muy interesante.

1.3 Objetivos

El objetivo del proyecto se basa en realizar un prototipo o sistema básico que permita gestionar la actividad principal de una pequeña empresa de transporte de paquetería. Dicho sistema se llamará Matroid (Mobile Application Transport Routing Optimized) y tendrá como ramas principales los siguientes puntos:

- **Aplicación Android**: La creación de una aplicación Android que será utilizada por los transportistas cuando estos estén en ruta. La aplicación deberá principalmente ser capaz de registrar el vehículo que van a utilizar, mostrar la información de la ruta, mostrar la información de la próxima parada y guiar mediante navegación giro a giro hasta esta, mostrar los datos del cliente, registrar las recogidas y las entregas de los paquetes y enviar la ubicación cada cierto tiempo al servidor para que este pueda realizar las correcciones oportunas de la ruta.

- **Servidor central**: La creación de un servidor central, mediante Java, que será capaz de ejecutar los cálculos necesarios para la creación y actualización de las rutas, recibir o enviar peticiones a los dispositivos móviles, realizar las acciones correspondientes a dichas peticiones y conectarse a un web service externo mediante el cual se apoyará para realizar los cálculos de las rutas.

- **Panel de control**: La creación de un panel de control que se basará en una página web, mediante PHP, que será capaz de buscar, visualizar, introducir, modificar o eliminar en los casos posibles y establecidos los paquetes, vehículos o rutas de forma fácil, ágil e intuitiva.

- **Página de seguimiento**: La creación de una página web, mediante PHP, que será capaz de mostrar a un usuario del servicio de transporte de paquetería los datos relativos a un paquete de forma clara, fácil e intuitiva mediante su identificador.

- **Base de datos**: La creación de una base de datos, mediante MySQL, que se encargará de almacenar los datos que serán consultados, creados, eliminados y modificados por el servidor central, el panel de control y la página de seguimiento.

El sistema Matroid se apoyará en un web service externo para realizar los cálculos más complejos, este será la interfaz de conexión con un servidor de cálculo externo formado por GPUs preparado y optimizado para realizar cálculos complejos. Tanto el web service como el servidor de cálculo están fuera del alcance del proyecto, aunque se definirá un cliente y un servidor web service de prueba que retornará una respuesta predefinida para demostrar la viabilidad del proceso.

2 Requisitos

En este apartado se definen los requisitos funcionales y no funcionales del sistema Matroid.

2.1 Requisitos funcionales

A continuación se definen de cada rama principal del proyecto los requisitos funcionales, que son aquellos que establecen el comportamiento del sistema.

2.1.1 Aplicación Android

Se definirán los requisitos funcionales correspondientes a la aplicación Android ejecutada en un dispositivo móvil Android por los transportistas.

1.1 Vinculación al vehículo

La aplicación permitirá y deberá de ser informada, para poder proseguir con las diferentes funciones, del vehículo que se va a utilizar a través de su identificador. De esta forma el dispositivo queda vinculado al vehículo, y todas las acciones que se hagan posteriormente desde el dispositivo serán en relación a este, hasta que se desvinculen.

1.2 Desvinculación del vehículo

La aplicación permitirá desvincular el dispositivo del vehículo, de forma que este quedará a la espera de que le sea introducido otro identificador para proseguir con las funciones.

1.3 Inicio o continuación de la ruta

La aplicación permitirá iniciar o continuar, si ya ha sido iniciada, la ruta asignada al vehículo.

1.4 Información ruta y parada

La aplicación mostrará la información correspondiente a la ruta que se está realizando, la próxima parada a realizar y los paquetes que se deben de entregar o recoger en la parada. Concretamente mostrará la información siguiente:

- Información ruta
  - Identificador
  - Fecha inicio programada
  - Fecha fin programada
- Información de la próxima parada
  - Identificador
  - Dirección
- Información de los paquetes de la parada
  - Identificador
  - Acción que se debe de realizar

- Resolución de la acción
- Titular al que se debe de dirigir
- Dirección
- Fecha inicio y fin de la ventana de tiempo definida para recoger o entregar el paquete

1.5. Visualización mapa

La aplicación permitirá visualizar e identificar en un mapa la posición exacta, a través de un marcador, de la próxima parada a realizar. El mapa debe dar la posibilidad de moverse a través de él y hacer zoom.

1.6. Navegación

La aplicación permitirá guiar al transportista mediante navegación giro a giro hasta la próxima parada a realizar.

1.7. Acciones parada

La aplicación debe de permitir definir las acciones que se llevan a cabo por cada paquete de la parada que se está realizando.

1.8. Siguiente parada

La aplicación debe de permitir, una vez definida todas las acciones de la parada en curso, obtener y mostrar todos los datos de la siguiente parada a realizar.

1.9. Retorno base

Cuando se haga una petición de *siguiente parada* y no queden más paradas a realizar en la ruta, la aplicación mostrará la dirección de la base a la que tiene que retornar y dará la posibilidad de visualizarla en el mapa y de realizar una navegación giro a giro como se han descrito anteriormente.

1.10. Fin ruta

La aplicación dará la posibilidad de indicar que la ruta en curso se ha finalizado una vez que se hayan realizado todas las paradas.

1.11. Ubicación

La aplicación deberá de informar, siempre que se esté ejecutando una ruta, cada cierto tiempo y/o distancia la ubicación del dispositivo al servidor central.

1.12. Actualización de la ruta

Siempre que se esté en modo de navegación giro a giro, en el caso de que la ruta haya sido actualizada y la actualización afecte a la parada a la que se está dirigiendo el transportista, el dispositivo podrá interrumpir la navegación y mostrar los datos de la parada actualizada. Además mostrará un mensaje de texto informando de la situación.

1.13. Conexión servidor central

La aplicación enviará la información sobre las acciones que se hagan al servidor. En caso de error al enviarla, la aplicación mostrará un mensaje de texto informando de la situación y no dejará proseguir hasta que se realice la transferencia de datos de forma correcta.

2.1.2 Servidor central

Se definirán los requisitos funcionales correspondientes al servidor central del sistema Matroid.

2.1 Crear rutas iniciales

El servidor será capaz de crear las rutas iniciales que se deben ejecutar en las próximas 8 horas, de la manera más óptima posible, y almacenarlas para que posteriormente los transportistas con los dispositivos las recorran. En el cálculo inicial se supondrá que todos los vehículos iniciarán la ruta desde la base y deben de volver a la misma. Los elementos de que se disponen para realizar el cálculo son todos los paquetes que se puedan entregar o recoger en las próximas 8 horas y todos los vehículos que estén disponibles.

2.2 Actualización rutas

El servidor será capaz de recalcular las rutas que se están realizando en ese momento para intentar optimizarlas. En el caso de que exista una ruta más optima, realizará una actualización para adaptar la ruta actual a la nueva. Si la actualización afecta a la parada que se está dirigiendo un transportista, el servidor enviará una notificación al dispositivo. En las actualizaciones se podrá añadir, eliminar o modificar de orden los paquetes, teniendo como efecto la creación, eliminación o modificación de las paradas que quedan por realizar.

2.3 Cálculo de distancias

El servidor será capaz de calcular las distancias y el tiempo estimado de recorrido entre los puntos necesarios para realizar el cálculo o actualización de la ruta. Por ejemplo, calcular la distancia y el tiempo estimado entre dos paquetes o entre el vehículo y su próxima parada.

2.4 Recibir y enviar información de dispositivos

El servidor será capaz de recibir la información enviada por los dispositivos y de enviarles información en los casos que sea necesario.

2.5 Ejecutar peticiones

El servidor será capaz de ejecutar las acciones asociadas a las peticiones que recibe de los dispositivos.

2.6 Conexión web service

El servidor será capaz de conectarse a un web service para enviar y recibir datos de este. La comunicación con dicho web service se realizará en formato XML, por lo tanto, el servidor debe de ser capaz de generar los XMLs con los datos necesarios y deberá de ser capaz de tratar los XMLs recibidos por el web service, realizando las acciones oportunas teniendo en cuenta la información recibida.

2.7 Conexión API Google Maps

El servidor será capaz de conectar con los servidores de Google Maps a través de su API para enviar y recibir datos. La información que se recibirá estará en formato XML, por lo tanto, el servidor debe de ser capaz de tratar los XMLs recibidos por la API.

2.8 Logs

En todos los procesos, el servidor escribirá un log definiendo el comportamiento realizado y los posibles errores encontrados.

2.1.3 Panel de control

Se definirán los requisitos funcionales correspondientes al panel de control utilizado por los administradores.

3.1 Login

El panel de control ofrecerá una pantalla donde el administrador deberá de identificarse correctamente para poder acceder. En el login se le requerirá el usuario y la contraseña.

3.2 Alta administrador

El panel de control deberá dar la posibilidad de dar de alta a administradores nuevos. Para realizar el proceso correctamente, se requerirá introducir los siguientes datos:

- Usuario
- Contraseña
- Nombre y apellido

Una vez dado de alta el administrador, se le redirigirá a la pantalla de login donde deberá de iniciar sesión.

3.3 Modificar administrador

El panel de control dará la posibilidad al administrador de modificar sus propios datos. Los datos que podrá modificar son:

- Nombre y apellido
- Contraseña

3.4 Eliminación administrador

El panel de control dará la posibilidad al administrador de eliminar su propio perfil. Una vez eliminado, será redirigido a la pantalla de login donde podrá iniciar sesión como otro administrador o dar de alta a un nuevo.

3.5 Búsqueda paquetes

El panel de control permitirá realizar una búsqueda por diferentes criterios de los paquetes, en caso de no introducir ningún criterio, se mostrarán todos los paquetes existentes. Los criterios por los que se podrá buscar son:

- Identificador
- Estado
- Titular de recogida
- Fecha de recogida
- Dirección de recogida
- Titular de entrega
- Fecha de entrega
- Dirección de entrega

En el caso de las fechas, se buscará todos los paquetes que incluyan la fecha introducida dentro de la ventana de tiempo de recogida o entrega correspondientemente.

3.6 Creación paquete

El panel de control permitirá la creación de paquetes nuevos para que sean introducidos en el sistema y contemplados para el cálculo de rutas. Para la creación de un paquete se deberá introducir los siguientes datos:

- Estado
- Titular de recogida
- Dirección de recogida
- Fecha inicio de recogida
- Fecha fin de recogida
- Titular de entrega
- Dirección de entrega
- Fecha inicio de entrega
- Fecha fin de entrega

3.7 Detalle paquete

El panel de control permitirá visualizar un detalle del paquete, mostrando toda la información almacenada de este. La información que se mostrará será:

- Identificador
- Estado
- Titular de recogida
- Dirección de recogida
- Fecha inicio de recogida
- Fecha fin de recogida
- Titular de entrega
- Dirección de entrega
- Fecha inicio de entrega
- Fecha fin de entrega
- Información del rastro

En la información del rastro, se informará de todas las acciones en las que se ha visto involucrado el paquete en las rutas que ha realizado.

3.8 Edición paquete

El panel de control permitirá la edición de los paquetes, los datos que se podrán editar son:

- Estado
- Titular de recogida
- Dirección de recogida
- Fecha inicio de recogida
- Fecha fin de recogida
- Titular de entrega
- Dirección de entrega
- Fecha inicio de entrega
- Fecha fin de entrega

3.9 Eliminación paquete

El panel de control permitirá la eliminación de los paquetes del sistema. Una vez eliminado el paquete, este no se podrá volver a recuperar.

3.10 Búsqueda rutas

El panel de control permitirá realizar una búsqueda de las rutas por diferentes criterios, en caso de no introducir ningún criterio se mostrarán todas las rutas existentes. Los criterios por los que se podrá buscar son:

- Identificador
- Fecha

En el caso de la fecha, se buscará todas las rutas que incluyan la fecha introducida dentro de la ventana de tiempo de las fechas programadas.

3.11 Detalle ruta

El panel de control permitirá visualizar un detalle de la ruta, mostrando toda la información almacenada de esta. La información que se mostrará será:

- Identificador
- Estado
- Identificador del vehículo que la realiza
- Fecha inicio programada
- Fecha fin programada
- Fecha inicio real
- Fecha fin real
- Paradas de la ruta

En la información de las paradas, se detallarán los paquetes con las correspondientes acciones que debe de realizar el transportista en ellas. Además, se informará de la hora y la resolución en caso de que la acción ya haya sido ejecutada.

3.12 Eliminación ruta

El panel de control permitirá la eliminación de las rutas siempre y cuando estas no se estén realizando. Una vez eliminada la ruta, esta no se podrá volver a recuperar. Cuando la ruta se elimine, los paquetes que estaban asignados a dicha ruta estarán de nuevo disponibles para que se les pueda asignar otra ruta, siempre y cuando cumplan los requerimientos necesarios.

3.13 Búsqueda vehículos

El panel de control permitirá realizar una búsqueda por diferentes criterios de los vehículos, en caso de no introducir ningún criterio, se mostrarán todos los vehículos existentes. Los criterios por los que se podrá buscar son:

- Identificador
- Matrícula
- Marca
- Modelo
- Color
- Disponibilidad

3.14 Creación vehículo

El panel de control permitirá la creación de vehículos nuevos para que sean introducidos en el sistema y contemplados para el cálculo de rutas. Para la creación de un vehículo se deberá introducir los siguientes datos:

- Matrícula
- Marca
- Modelo
- Color
- Disponibilidad

3.15 Detalle vehículo

El panel de control permitirá visualizar un detalle del vehículo, mostrando toda la información almacenada de este. La información que se mostrará será:

- Matrícula
- Marca
- Modelo
- Color
- Disponibilidad
- Ruta actual que está realizando
- Dispositivo asignado
- Última dirección aproximada conocida
- Última latitud y longitud conocida
- Fecha de los últimos datos de ubicación

3.16 Edición vehículo

El panel de control permitirá la edición de los vehículos, los datos que se podrán editar son:

- Matrícula
- Marca
- Modelo
- Color
- Disponibilidad

La disponibilidad del vehículo solamente será contemplada la próxima vez que se generen nuevas rutas.

3.17 Eliminación vehículo

El panel de control permitirá la eliminación de los vehículos siempre y cuando no estén realizando una ruta. En el caso de eliminar el vehículo, todas las rutas que tenia asignadas y que no se hayan realizado, serán eliminadas automáticamente. Una vez eliminado el vehículo, este no se podrá volver a recuperar.

3.18 Resultados búsqueda

Los resultados de la búsqueda de todos los tipos de elementos (paquete, ruta y vehículo) serán mostrados en una lista paginada de 10 filas. En cada fila se mostrará una información resumida y la posibilidad de acceder al detalle del elemento.

3.19 Validación formulario

Todos los formularios validarán los datos antes de realizar la acción correspondiente. En caso de error, se mostrará un mensaje para que se pueda corregir.

3.20 Mapas

Toda dirección, ruta o ubicación del vehículo mostrada por el panel de control se verá acompañada por un mapa que indicará, con un marcador o marcadores, la posición del elemento. Además, en caso de que la dirección se modifique, el mapa deberá de ir actualizándose mientras se escribe la nueva dirección.

El mapa deberá de dar la posibilidad de moverse a través de él y hacer zoom. Además, se podrá visualizar en modo mapa, satélite o a pie de calle.

3.21 Fechas

Todas las fechas que se introducen o se modifican en el panel de control se verán acompañadas de un pop-up, que muestra un calendario, para ayudar a indicar la fecha deseada de forma más cómoda y fácil.

2.1.4 Página de seguimiento

Se definirán los requisitos funcionales correspondientes a la página de seguimiento. Está página tiene un carácter más público y será utilizada por los usuarios de la empresa de paquetería.

4.1 Introducción identificador paquete

La página de seguimiento permitirá introducir el identificador de un paquete para visualizar los datos disponibles. En caso de no existir dicho paquete, se mostrará un mensaje de alerta.

4.2 Visualización del paquete

Una vez introducido el identificador del paquete, si este existiera, la página de seguimiento mostraría la información correspondiente. La información que se mostrará es la siguiente:

- Identificador
- Estado
- Titular de recogida
- Dirección de recogida
- Fecha inicio de recogida
- Fecha fin de recogida
- Titular de entrega
- Dirección de entrega
- Fecha inicio de entrega
- Fecha fin de entrega
- Información del rastro

En la información del rastro, se informará de todas las acciones en las que se ha visto involucrado el paquete en las rutas que ha realizado.

Las direcciones de recogida y entrega se verán acompañadas por un mapa que indicará, junto a un marcador, la ubicación de la recogida o entrega.

2.2 Requisitos no funcionales

Los requisitos no funcionales son todos aquellos que están implícitos en el sistema y que afectan colateralmente al buen funcionamiento.

1. Roles

A continuación se definen los roles implicados en el sistema Matroid:

- **Transportista:** Empleado de la compañía de paquetería que realiza las rutas con la ayuda de la aplicación Android del sistema Matroid.
- **Administrador:** Empleado de la compañía de paquetería que puede acceder al panel de control del sistema Matroid.
- **Usuario del servicio:** Usuario de la compañía de paquetería que utiliza el sistema Matroid.

2. Estados de un paquete

Un paquete tiene uno de los siguientes estados:

- **Sin asignar recogida:** El paquete no se le ha asignado ninguna ruta para ser recogido.
- **Pendiente de recoger:** El paquete tiene asignado una ruta para ser recogido.
- **Recogido:** El paquete ha sido recogido correctamente.
- **Almacén:** El paquete está en el almacén y no se le ha asignado ninguna ruta para ser entregado.
- **En ruta:** El paquete tiene asignado una ruta para ser entregado.
- **Entregado:** El paquete ha sido entregado correctamente.

En el siguiente diagrama de flujos se muestran las transiciones entre los diferentes estados:

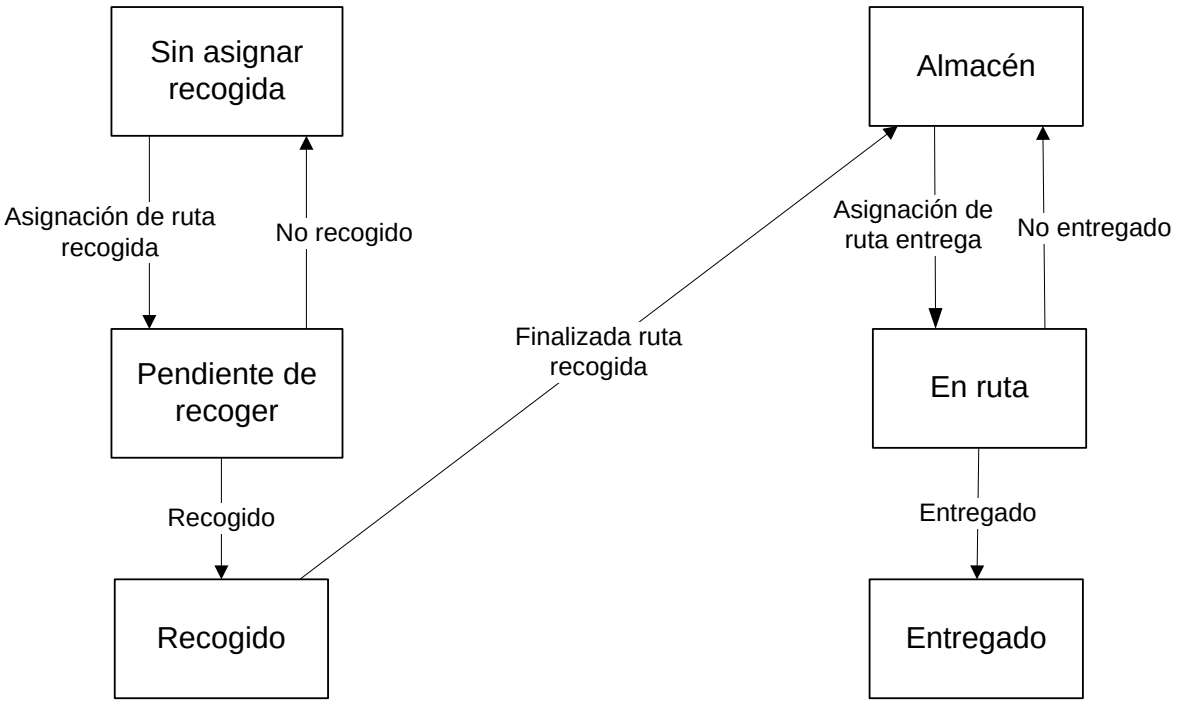


Fig. 1 Diagrama estados paquete

3. Estados de una ruta

Una ruta tiene uno de los siguientes estados:

- **Sin realizar:** La ruta aún no se ha iniciado.
- **Realizando:** La ruta se está realizando en este momento.
- **Finalizada:** La ruta se ha finalizado.

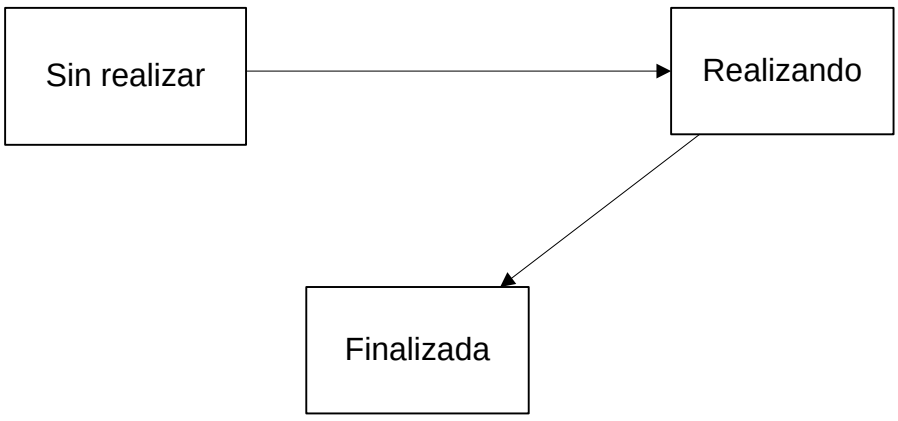


Fig. 2 Diagrama estados ruta

4. Acciones parada-paquete

Las acciones que se pueden realizar sobre los paquetes en cada parada son:

- **Recoger o entregar:** Recoger o entregar el paquete.
- **No realizado:** En el caso de que no se haya podido recoger o entregar el paquete.

5. Identificador dispositivo móvil

El identificador de todo dispositivo móvil será el mismo que nos proporciona el sistema Google Cloud Messaging al registrarlo. La aplicación no podrá proseguir sin la obtención de este identificador.

6. Relación vehículo-dispositivo

Un vehículo sólo puede tener asignado un dispositivo al mismo tiempo. Si se quiere asignar otro dispositivo al vehículo, primero se debe de desvincular el dispositivo actual y posteriormente vincular el nuevo dispositivo.

7. Versión Android

La aplicación será compatible con Android 2.2 o superiores y que soporten OpenGL ES 2.0

8. Reintento conexión

En el caso de que haya un error de comunicación entre el servidor central y la aplicación, o viceversa, ambos tendrán la capacidad de reintentar varias veces la conexión antes de mostrar el mensaje de error correspondiente.

9. Cálculo rutas

Para realizar el cálculo óptimo de rutas se utilizará la ayuda de un servidor de cálculo externo de ámbito matemático. Dicho servidor será capaz de realizar el cálculo óptimo de las rutas, según los datos enviados, y retornarlos.

Como hemos indicado el servidor de cálculo es de ámbito matemático, por lo tanto no hace referencia a vehículos, rutas o paquetes. Simplemente se le envían datos del estilo: un número N de puntos, cada uno hace referencia a un paquete y contiene su ventana de tiempo y las distancias y los tiempos aproximados con el resto de puntos; y un número M de identificadores, haciendo referencia a los vehículos disponibles para realizar las rutas.

Tanto la información que se envía como la que se recibe se transmitirá mediante un web service y deberá de seguir un formato especificado en XML (el formato está definido en el anexo).

10. Disponibilidad vehículo

El campo que indica si un vehículo está disponible o no, es un campo manual que puede modificar el administrador desde el panel de control. Dicho campo indica la disponibilidad del vehículo para tenerlo en cuenta, o no, en los futuros cálculos de rutas iniciales. En ningún caso indica si el vehículo está realizando una ruta o no.

11. Cálculo dirección parada

La dirección de la parada será obtenida de algún paquete que esté asignado a esta. Se sobreentiende que todos los paquetes de una parada se pueden entregar o recoger sin volver a subir al vehículo porque su dirección es muy próxima.

12. Cálculo distancias

Para realizar el cálculo de la distancia y el tiempo aproximado entre dos puntos, necesario para preparar los datos a enviar al servidor de cálculo, utilizaremos la API de matriz de distancia de Google.

13. Cálculo coordenadas

Para realizar el cálculo de una dirección a coordenadas o viceversa, se utilizará la API de codificación geográfica de Google.

14. Recoger o entregar

Un paquete no se podrá entregar y recoger en la misma ruta.

15. Visualización mapas

Para visualizar los mapas en el panel de control y en la página de seguimiento se utilizará la API de Google Maps, mientras que para visualizar los mapas en la aplicación Android se utilizará la API Google Maps Android.

16. Notificaciones push

Cuando el servidor central necesite enviar datos a algún dispositivo móvil, se utilizará la tecnología de notificaciones push. Para utilizar dicha tecnología utilizaremos el servicio Google Cloud Messaging for Android (GCM).

17. Formato envío información servidor central

La información enviada desde el servidor central a los dispositivos móviles será formateada con JSON.

18. Control petición dispositivo

Cuando un dispositivo móvil realice una petición al servidor central, este realizará una comprobación de forma que únicamente el dispositivo vinculado al vehículo podrá realizar esa petición en su nombre.

19. Repetición usuarios

No podrá haber dos administradores que tengan el mismo nombre de usuario.

20. Validación y seguridad de los campos

Tanto en el panel de control como en la página de seguimiento, todos los campos en los que se permite introducir datos serán validados y se les aplicará filtros para evitar la inyección de código JavaScript y SQL.

21. Codificación contraseñas

Las contraseñas de los administradores serán encriptadas con los algoritmos SHA1 y MD5, además de aplicarles un SALT como prefijo y sufijo antes de ser almacenados.

22. Amigable

La aplicación Android, el panel de control y la página de seguimiento deberán mostrar la información de forma bien distribuida y que el acceso a todas las funcionalidades sea intuitivo y simple.

23. Formato de los datos

A continuación, se definirán el formato de los diferentes campos del sistema Matroid:

- Identificador paquete, vehículo, ruta y usuario: Número mayor que cero.
- Estado paquete: Sin asignar recogida, Pendiente de recoger, Recogido, Almacén, En ruta y Entregado.
- Acción parada: Recoger y Entregar.
- Resolución parada: Entregado, Recogido y No realizado.
- Ruta: Sin realizar, Realizándose y Realizada.
- Fechas: dd/MM/yyyy hh:mm
- Nombre de usuario:
    - Sólo puede contener minúsculas, números, puntos y guiones bajos.
    - Debe de empezar y finalizar con una minúscula o un número.
    - Debe de tener una longitud entre 6 y 30 caracteres.
- Contraseña: Debe de tener una longitud entre 6 y 30 caracteres.
- Nombre y apellido: La longitud máxima es de 50 caracteres.
- Matrícula: Cadena alfanumérica de tamaño máximo 8 caracteres.

24. Lenguajes, tecnologías, framework y APIs

Los lenguajes que se utilizarán para implementar el sistema Matroid son:

- Java1.6: Implementación de la aplicación Android y el servidor central.
- HTML5: Implementación del panel de control y página de seguimiento por el lado del cliente.
- JavaScript y JQuery: Implementación del panel de control y página de seguimiento por el lado del cliente.
- PHP5: Implementación del panel de control y página de seguimiento por el lado del servidor y cliente.
- SQL: Implementación de la base de datos.

Los formatos de transferencia que se utilizarán son:

- XML: Transferencia de datos entre servidor central y servidor de cálculo.
- JSON: Transferencia de datos entre el servidor central y los dispositivos móviles.

Las tecnologías que se utilizarán para implementar el sistema Matroid son:

- Sistema operativo Android: Para la ejecución de la aplicación Android.
- JVM: Para la ejecución de los procesos del servidor central.
- Tomcat6: Para la recepción de mensajes enviados por los dispositivos móviles.
- Apache2: Servidor web para el panel de control y la página de seguimiento.
- MySQL: Gestor de base de datos.

Los framework que se utilizarán para implementar el sistema Matroid son:

- CodeIgniter: Para la implementación del panel de control y la página de seguimiento por el lado del servidor y cliente.
- Bootstrap: Para la implementación del panel de control y la página de seguimiento por el lado del cliente.

Las APIs que se utilizarán para implementar el sistema Matroid son:

- Google Cloud Messaging for Android (GCM): Para enviar notificaciones push a los dispositivos móviles.
- Google Maps API v3: Para visualizar los mapas en el panel de control y la página de seguimiento.
- Google Maps Android API v2: Para visualizar los mapas en la aplicación Android.
- API de matriz de distancia de Google: Para calcular la distancia entre dos puntos.
- API de codificación geográfica de Google: Para calcular las coordenadas de una dirección o viceversa

3 Diseño UML

3.1 Visión general

A continuación, se explicará desde un punto de vista general el funcionamiento del sistema Matroid sin entrar en detalles técnicos que serán explicados más adelante.

En primer lugar, se describirá el funcionamiento de la característica principal del sistema que abarca la creación, actualización y realización de las rutas:

1. Cada 8 horas el servidor central calculará las rutas que se deben de realizar. Para realizar el cálculo, el servidor obtendrá todos los paquetes que se puedan entregar o recoger en las próximas 8 horas y todos los vehículos disponibles.
2. Los transportistas, con sus dispositivos, se vincularán al vehículo que van a utilizar e irán obteniendo información de la ruta que deben de realizar.
3. Cuando el transportista recoja o entregue un paquete lo indicará en la aplicación.
4. Mientras el transportista esté en ruta, el dispositivo irá enviando su ubicación al servidor central.
5. Cuando el servidor central reciba una ubicación por parte del dispositivo, realizará un recálculo de la ruta para intentar encontrar una ruta más óptima. En el caso de encontrarla, modificará la ruta actual para adaptarla a la nueva ruta.
6. En el caso de que la actualización afecte a la parada que se está dirigiendo el transportista, el servidor enviará una notificación al dispositivo para que avise al transportista de la nueva ruta.
7. Una vez el trasportista realice todas las paradas de la ruta, la aplicación le informará como volver a la base. Una vez que llegue a la base, el transportista indicará a la aplicación que ha finalizado la ruta.

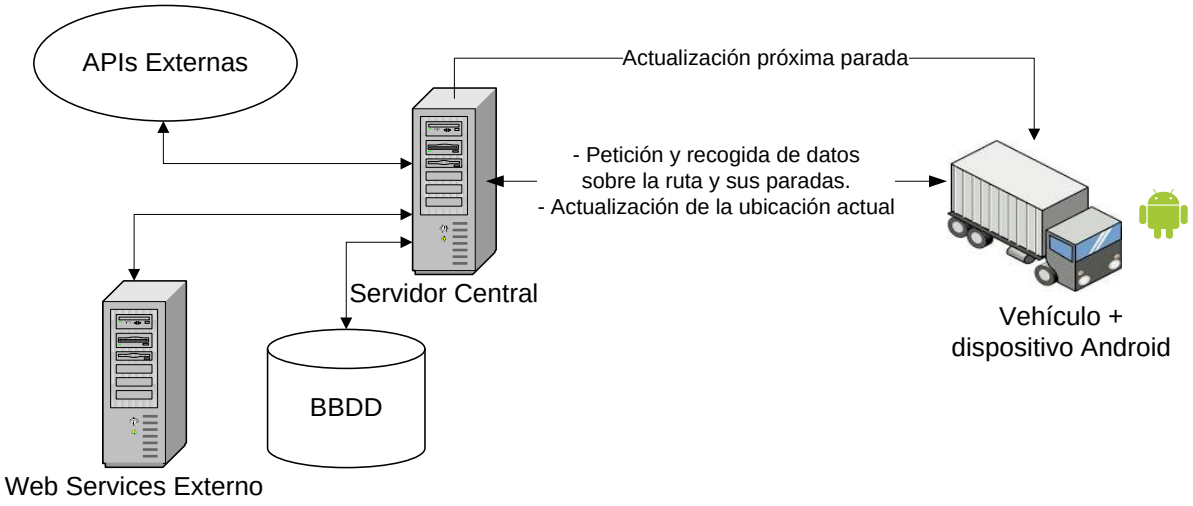


Fig. 3 Visión general principal

En segundo lugar, se describirá el funcionamiento del panel de control utilizado por los administradores para consultar, crear, editar y eliminar los elementos del sistema Matroid:

1. El administrador accede a la página de panel de control.
2. El administrador se identificará con su usuario y contraseña.
3. Si el usuario y contraseña son correctos, el administrador accederá al panel de control del sistema Matroid.

En último lugar, se describirá el funcionamiento de la página de seguimiento que es utilizada por los usuarios de la empresa de paquetería:

1. El usuario accede a la página de seguimiento.
2. El usuario introduce el identificador del paquete.
3. Si el identificador es correcto, se mostrarán los datos asociados al paquete.

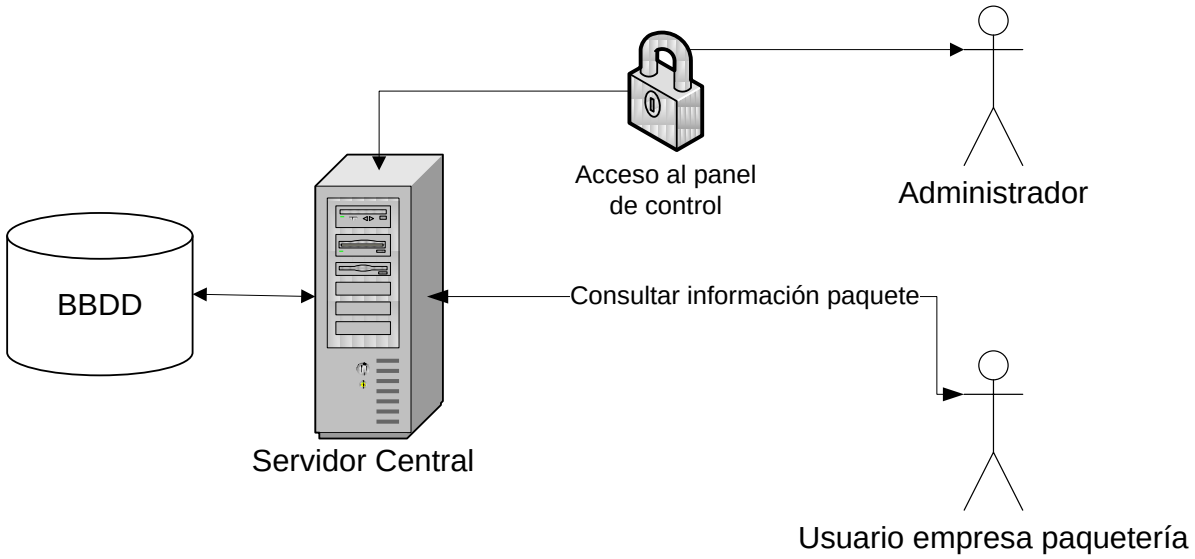


Fig. 4 Visión general web

3.2 Casos de uso

En este apartado se definirán los diferentes casos de usos según cada actor que interactúa con el sistema Matroid.

3.2.1 Aplicación Android

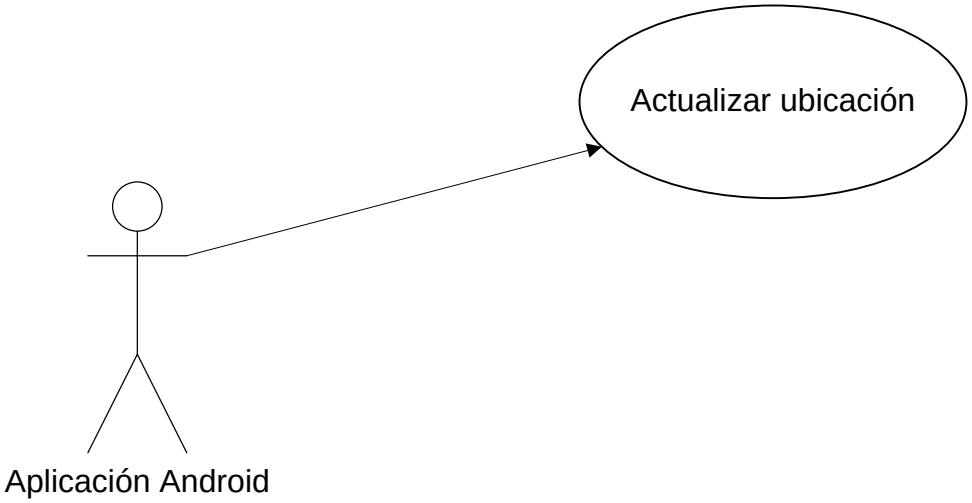


Fig. 5 Caso de uso aplicación Android

3.2.2 Servidor central

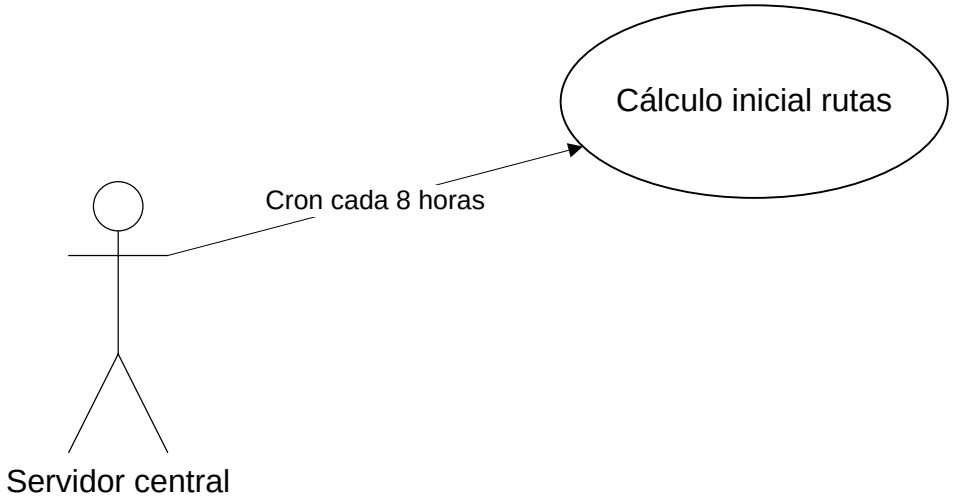


Fig. 6 Caso de uso servidor central

3.2.3 Transportista

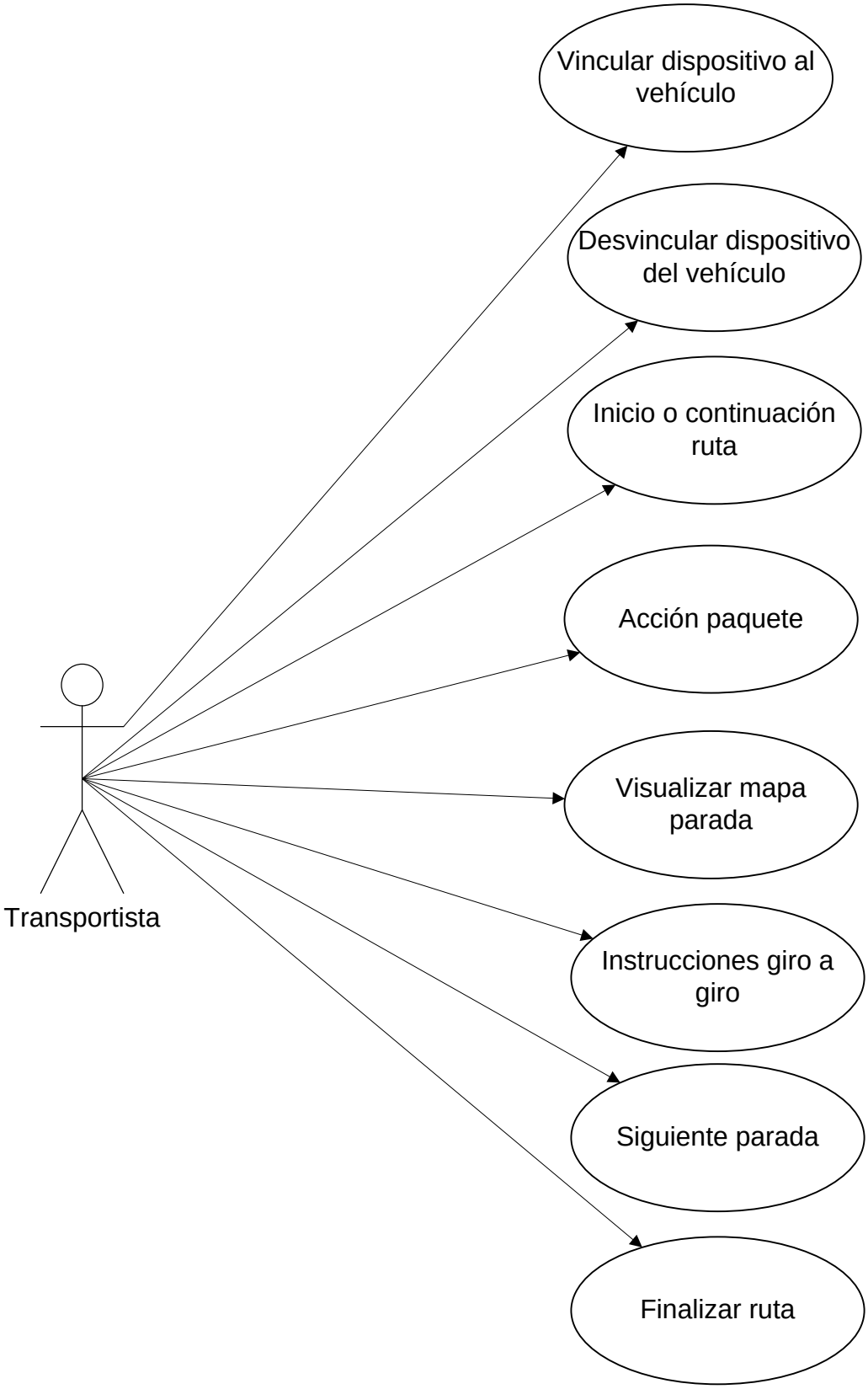


Fig. 7 Caso de uso transportista

3.2.4 Administrador

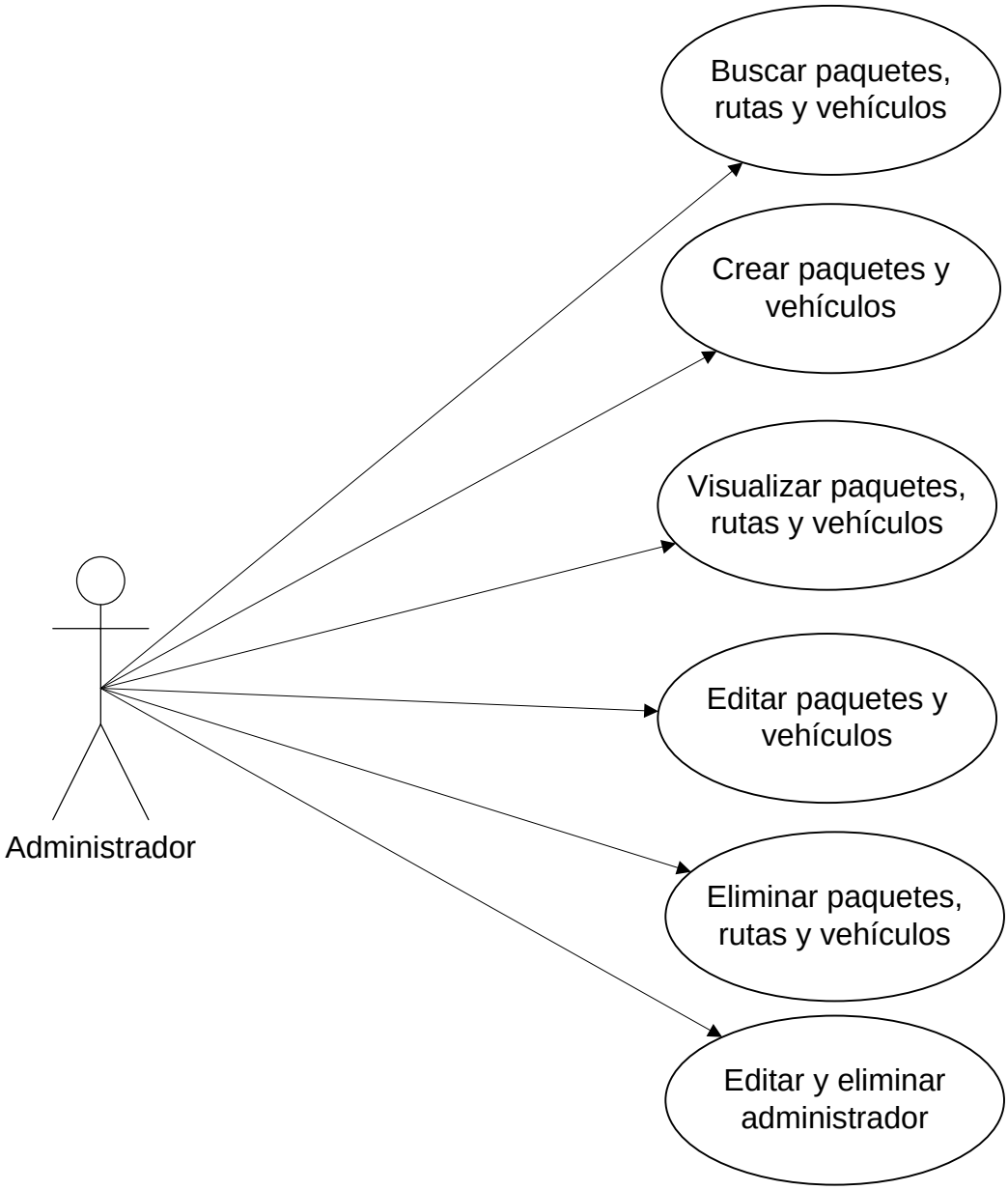


Fig. 8 Caso de uso administrador

3.2.5 Cliente de la empresa de paquetería

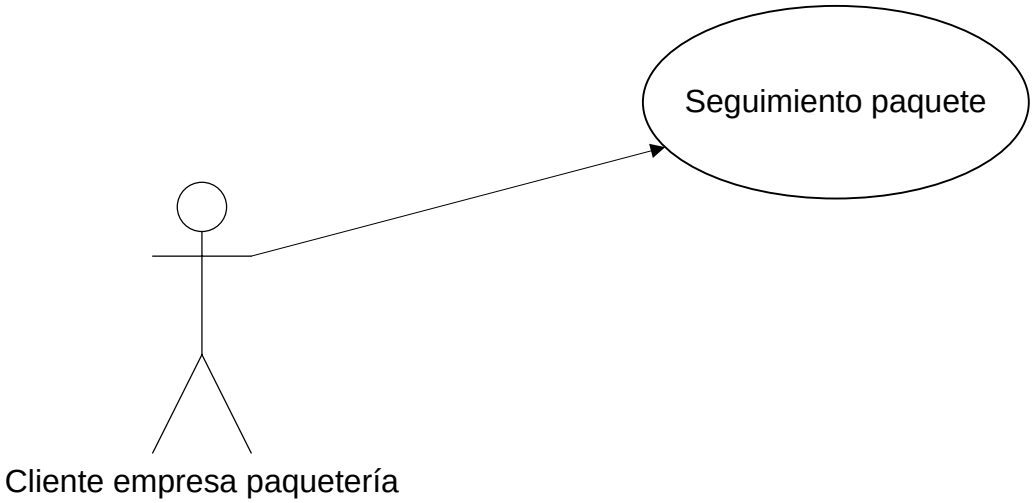


Fig. 9 Caso de uso cliente empresa paquetería

3.3 Diagramas de secuencia

En este apartado se definirán los diagramas de secuencia del sistema Matroid.

3.3.1 Cálculo inicial rutas

El cálculo inicial de las rutas se ejecuta, en un principio, automáticamente cada 8 horas a través de un cron. El proceso ejecuta los siguientes pasos:

1. Establece la fecha y hora de inicio y fin previstas, que serán aplicadas a todas las rutas que se vayan a crear. La hora de inicio es la hora actual, en el momento de la ejecución, y la hora final es la misma más 8 horas.
2. Obtiene todos los vehículos disponibles.
3. Obtiene todos los paquetes posibles para las rutas. Los paquetes disponibles son aquellos que se tienen que entregar o recoger dentro del plazo de duración prevista para la ruta.
4. Realiza el cálculo de las distancias entre todos los paquetes incluyendo la base. Con estos resultados genera el XML que será enviado al servidor externo.
5. Envía el XML al web service para que calcule las rutas óptimas y espera el XML de respuesta.
6. Trata el XML de respuesta creando las rutas que se han definido en él y modificando el estado de los paquetes que se ven afectados.

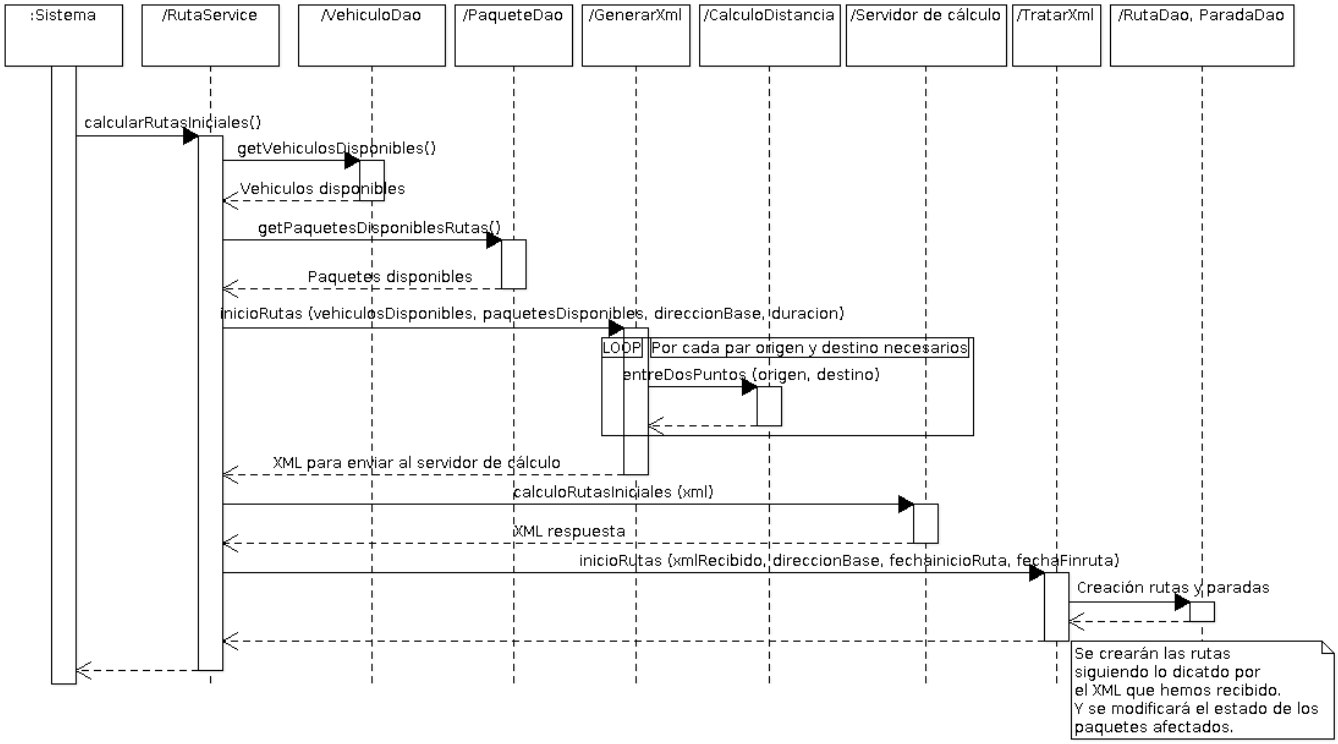


Fig. 10 Diagrama secuencia cálculo inicial rutas

3.3.2 Vinculación dispositivo al vehículo

El proceso de vinculación de un dispositivo al vehículo seguirá los siguientes pasos:

1. El transportista introduce el identificador X del vehículo.
2. El dispositivo envía al servidor la petición de vinculación con el vehículo X.
3. El servidor comprobará que el dispositivo no esté vinculado a otro vehículo. En caso de que así sea, lo desvinculará.
4. Comprobará que el vehículo X no esté vinculado a otro dispositivo.
5. Vinculará el dispositivo al vehículo X.
6. Si el proceso se ha ejecutado correctamente, el dispositivo guardará a que vehículo está vinculado, en este caso al vehículo X, y mostrará el menú principal.
7. En caso contrario, mostrará una alerta al transportista.

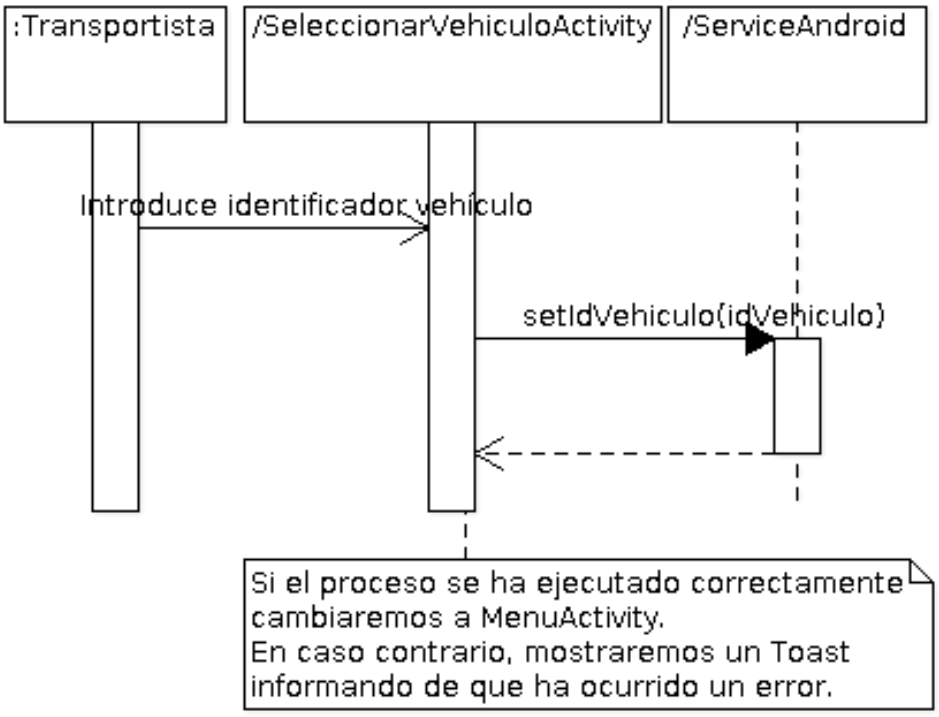


Fig. 11 Diagrama secuencia vinculación vehículo

3.3.3 Desvinculación dispositivo móvil a vehículo

El proceso de desvinculación de un dispositivo es un proceso similar al de la vinculación. Concretamente sigue los siguientes pasos:

1. El dispositivo envía al servidor la petición de vinculación con el vehículo -1.
2. El servidor desvinculará el dispositivo del vehículo al que está vinculado actualmente.
3. Si el proceso se ha ejecutado correctamente, el dispositivo eliminará la vinculación con el vehículo y mostrará la vista para escoger un nuevo vehículo a vincular.
4. En caso contrario, mostrará una alerta al transportista.

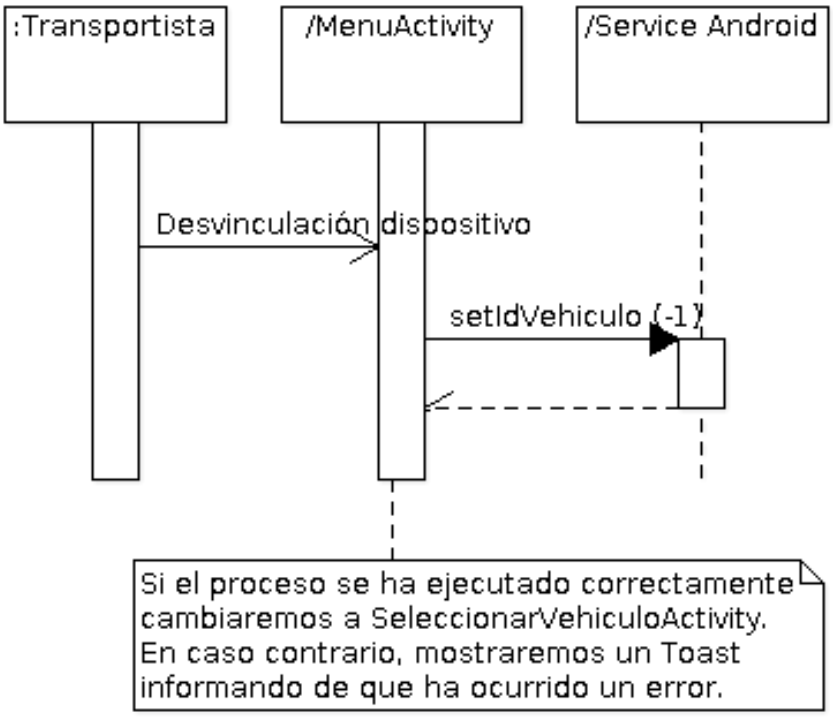


Fig. 12 Diagrama secuencia desvinculación vehículo

3.3.4 Inicio o continuación de la ruta

El proceso de inicio o continuación de la ruta es el mismo ya que el dispositivo solamente tiene conocimiento de la siguiente parada que va a realizar. El proceso sigue los siguientes pasos:

1. El transportista indica que quiere iniciar o continuar la ruta.
2. El dispositivo realiza una petición al servidor para obtener la siguiente parada a realizar de la ruta asignada.
3. El servidor obtiene la siguiente parada a realizar de la ruta asignada.
4. El servidor monta un token con la información que debe de ser enviada al dispositivo.
5. El servidor codifica el token en JSON y la envía al dispositivo.
6. Si existe la siguiente parada, el dispositivo la almacena, activa la localización del dispositivo y muestra el menú de parada.
7. En caso de que no existiera, muestra un aviso al transportista.

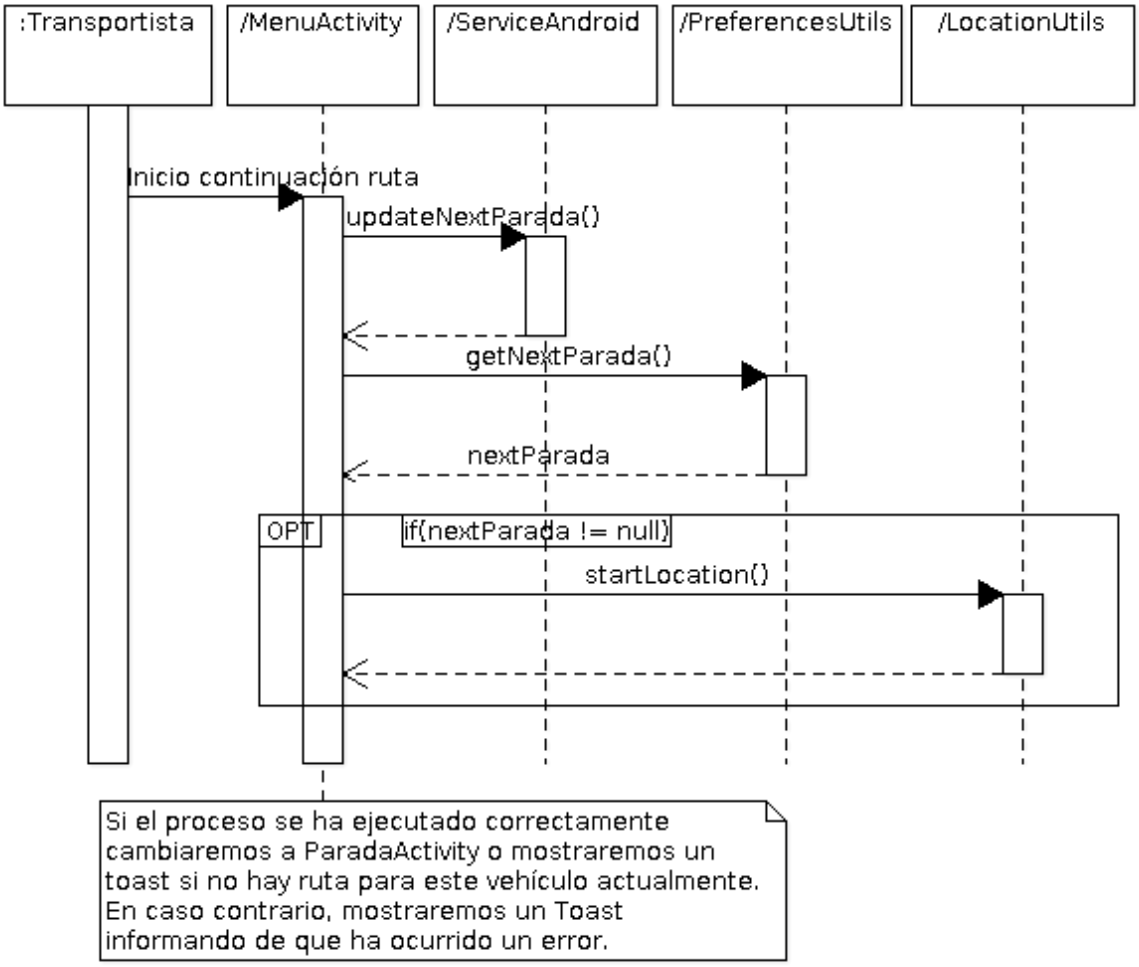


Fig. 13 Diagrama secuencia inicio o continuación ruta

3.3.5 Acción Paquete

El proceso de acción de paquete se ejecutará siempre que un transportista indique que ha recogido un paquete, lo ha entregado o que no ha podido realizar la acción. El proceso sigue los siguientes pasos:

1. El transportista indica la acción del paquete (recogido, entregado o no realizado).
2. El dispositivo realiza una petición con la información.
3. El servidor registra la acción realizada.
4. Tanto si el proceso se ejecuta correctamente como si falla, el dispositivo mostrará una alerta indicando la resolución de la petición.

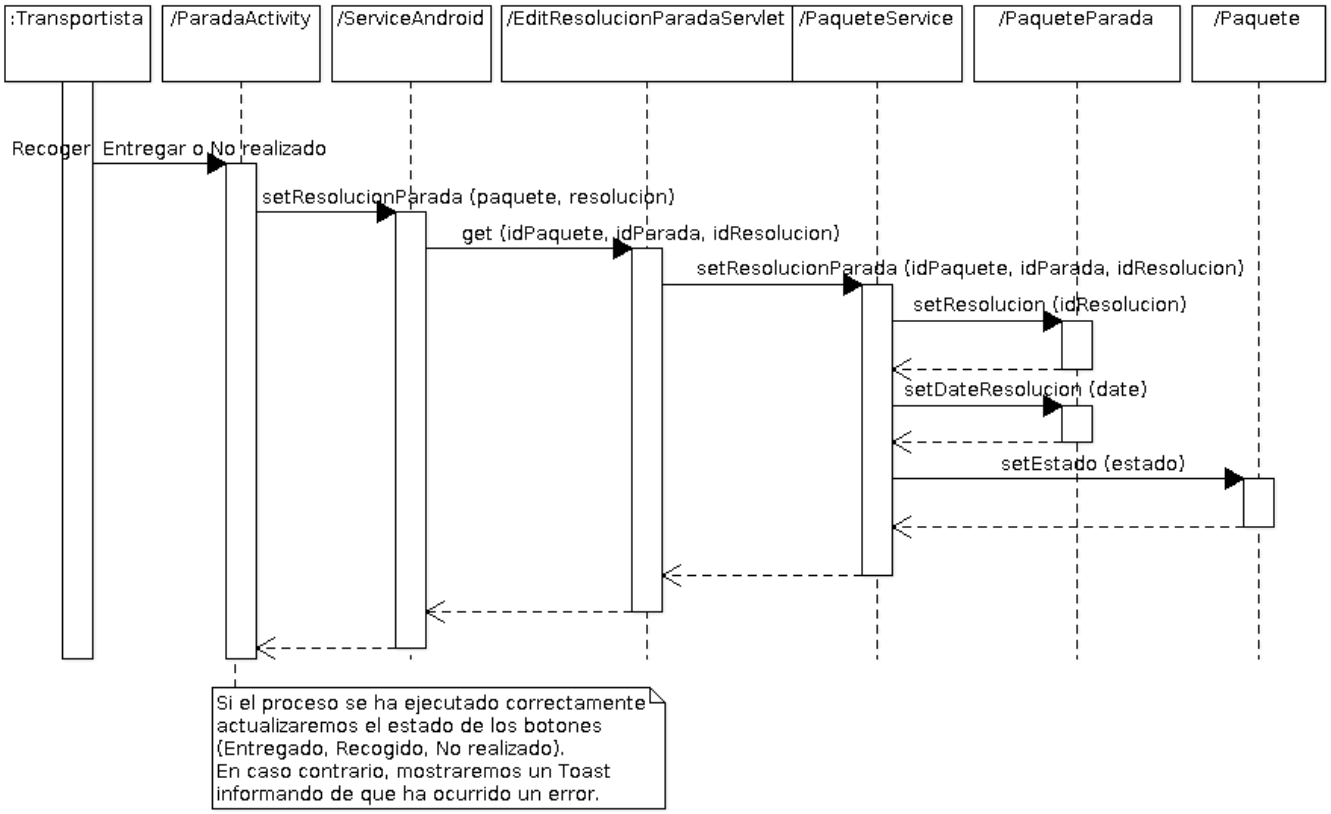


Fig. 14 Diagrama secuencia acción paquete

3.3.6 Siguiente parada

Al finalizar una parada, el transportista indicará que quiere realizar la siguiente y entonces se ejecutará este proceso:

1. El transportista indica que quiere realizar la siguiente parada.
2. El dispositivo envía la petición al servidor.
3. El servidor obtiene la siguiente parada de la ruta que tenga algún paquete en el que no se haya informado de la recogida, entrega o la no realización de la acción.
4. El servidor monta un token con la información que debe de ser enviada al dispositivo.
5. El servidor codifica el token en JSON y la envía al dispositivo.
6. El dispositivo muestra la información obtenida de la siguiente parada o la información de retorno a base si no hay más paradas a realizar en la ruta.
7. En caso de error en el proceso, el dispositivo mostrará una alerta.

Cabe destacar que si el transportista no ha realizado todas las acciones de los paquetes de la parada actual, al solicitar la siguiente parada se le mostrará la misma parada.

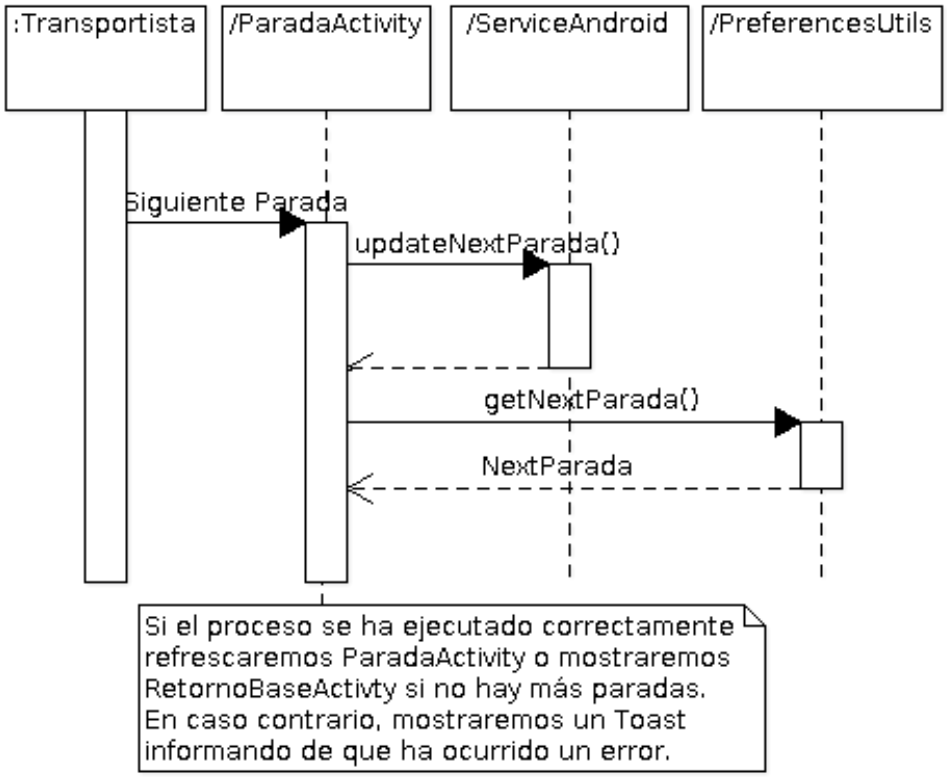


Fig. 15 Diagrama secuencia siguiente parada

3.3.7 Actualización ubicación

Cuando el transportista se encuentra realizando una ruta, cada cierto tiempo y/o distancia se realizará este proceso:

1. El dispositivo envía una petición de actualización de ubicación del vehículo de manera automática al servidor, indicando además si el transportista está preparado para recibir una actualización. Se considera que está preparado para recibir una actualización, cuando se esté ejecutando la navegación giro a giro hacia la siguiente parada o hacia la base.
2. Para evitar una saturación, el servidor aplicará un filtro de tiempo entre una petición y otra del mismo dispositivo.
3. Si supera el filtro, el servidor obtiene la ubicación y la almacena.
4. En el caso de que esté preparado para recibir una actualización, se lanzará un nuevo thread para intentar encontrar una ruta más óptima.

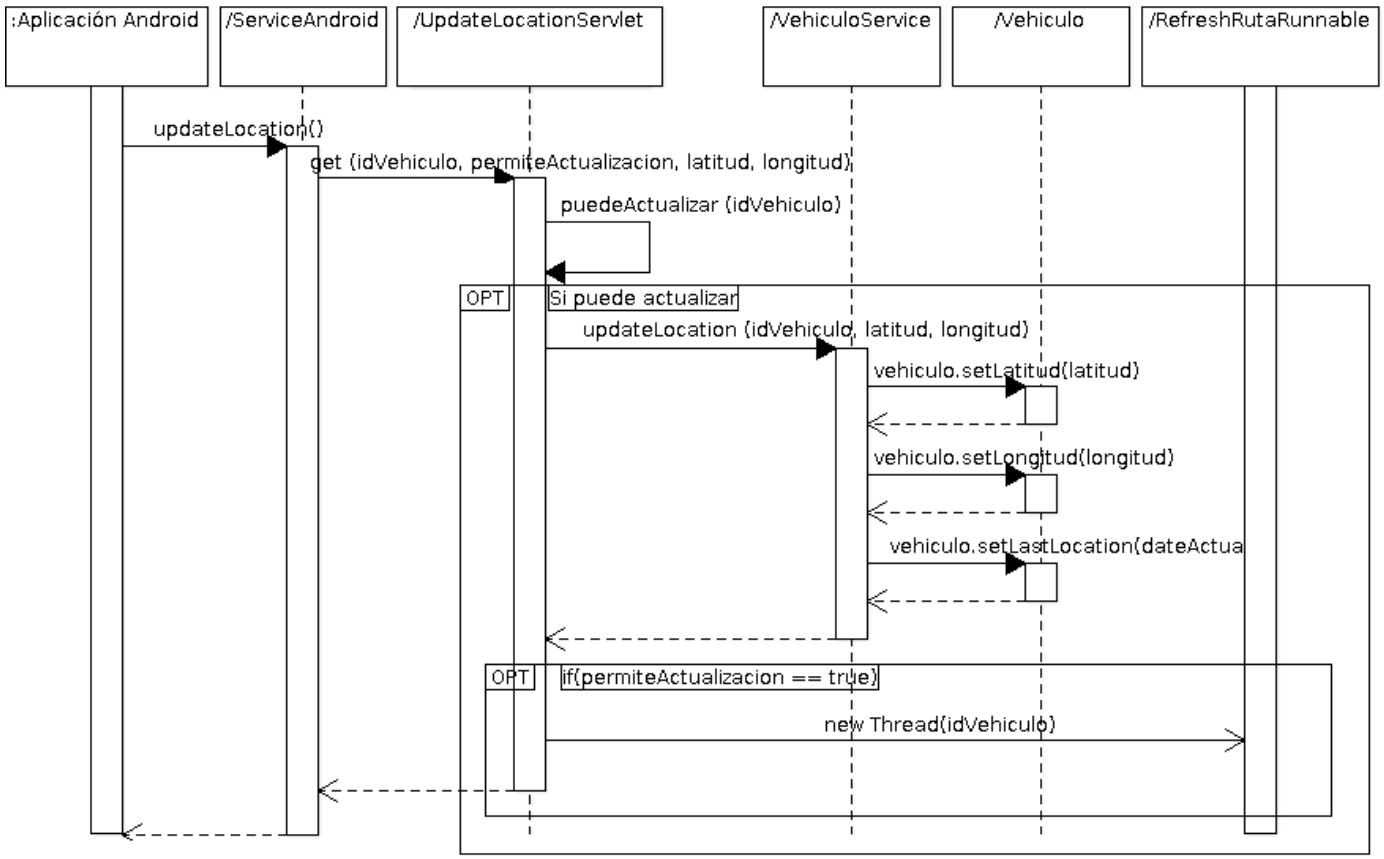


Fig. 16 Diagrama secuencia actualización ubicación

3.3.8 Actualización ruta

El cálculo de actualización de la ruta se ejecuta cada vez que el dispositivo indique su ubicación y haya superado el filtro del servidor. En ese caso se realiza el siguiente proceso:

1. Se obtiene todos los paquetes de las paradas que aún no se han realizado.
2. Se obtienen todas las recogidas que pueden ser añadidas a la ruta.
3. Establece el estado de los paquetes temporalmente como si no pertenecieran a la ruta.
4. Calcula todas las distancias entre los paquetes incluyendo la posición actual y la base; y con esta información genera el XML que será enviado al servidor.
5. Envía el XML al web service para que recalcule la ruta óptima y espera el XML de respuesta.
6. Trata el XML de la respuesta comparando la ruta que indica con la parte de la ruta actual que queda por realizar. En caso de encontrar diferencias, actualiza la ruta actual para adaptarse a lo indicando por el XML.
7. Restablece el estado de los paquetes que siguen o se han añadido a la ruta.
8. En caso de que la parada a la que se dirige el transportista se viera afectada por la actualización, el servidor central mandaría una notificación push al dispositivo.
9. Si el dispositivo recibe la notificación push, realizará una petición para obtener los datos de la parada actualizada (petición de siguiente parada). Una vez obtenido estos datos, los mostrará y añadirá una alerta indicando que la parada se ha modificado.

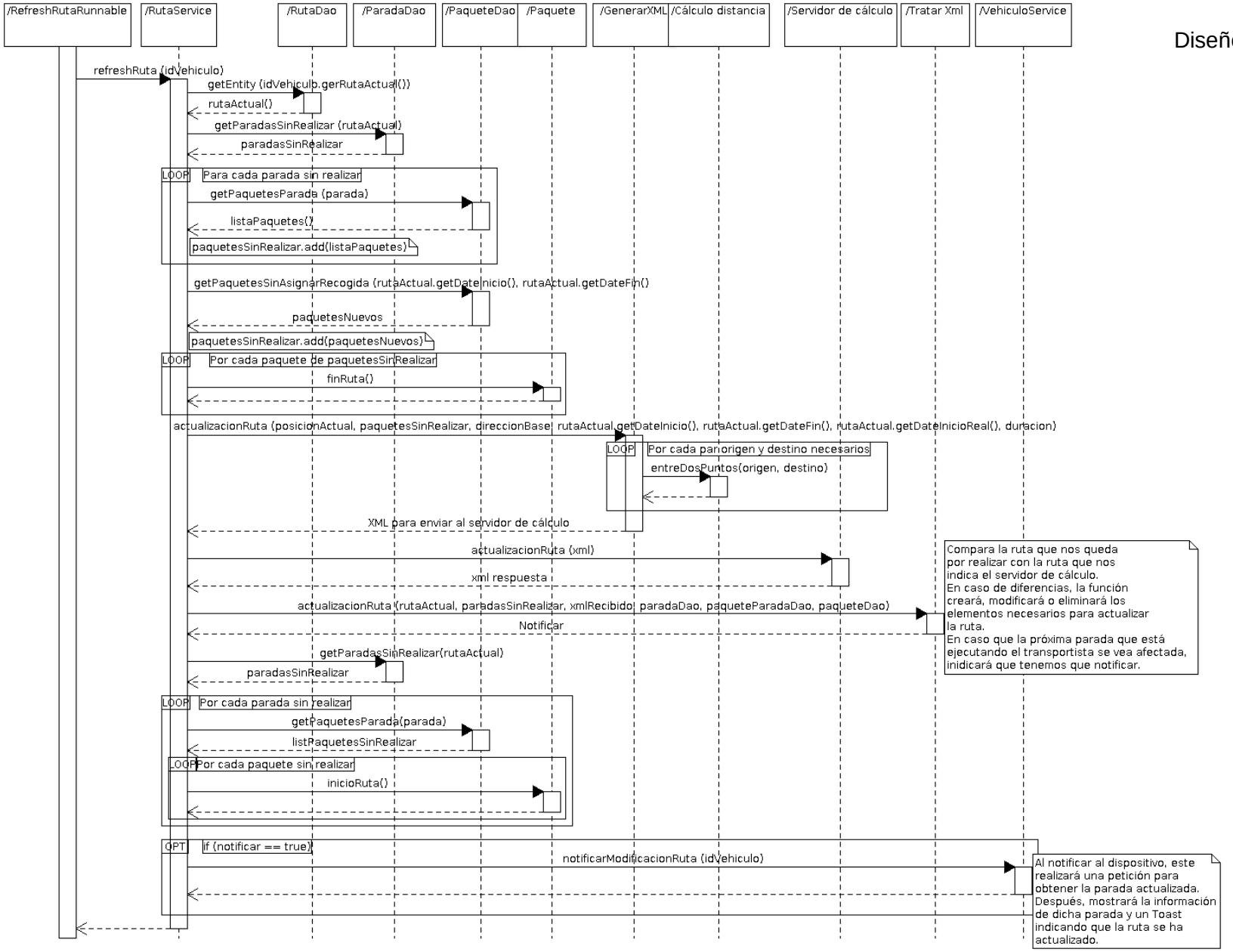


Fig. 17 Diagrama secuencia actualización ruta

3.3.9 Finalización ruta

Cuando un transportista llega a la base después de completar la ruta, debe indicar que ha finalizado la ruta. En ese momento se ejecuta el siguiente proceso:

1. El transportista indica que ha finalizado la ruta.
2. El dispositivo envía la petición al servidor.
3. El servidor modificará todos los paquetes de la ruta en concordancia con la finalización de la ruta.
4. El servidor guardará la hora real de finalización de la ruta.
5. Si el proceso se ha ejecutado correctamente, mostrará el menú principal de la aplicación.
6. En caso contrario, mostrará una alerta.

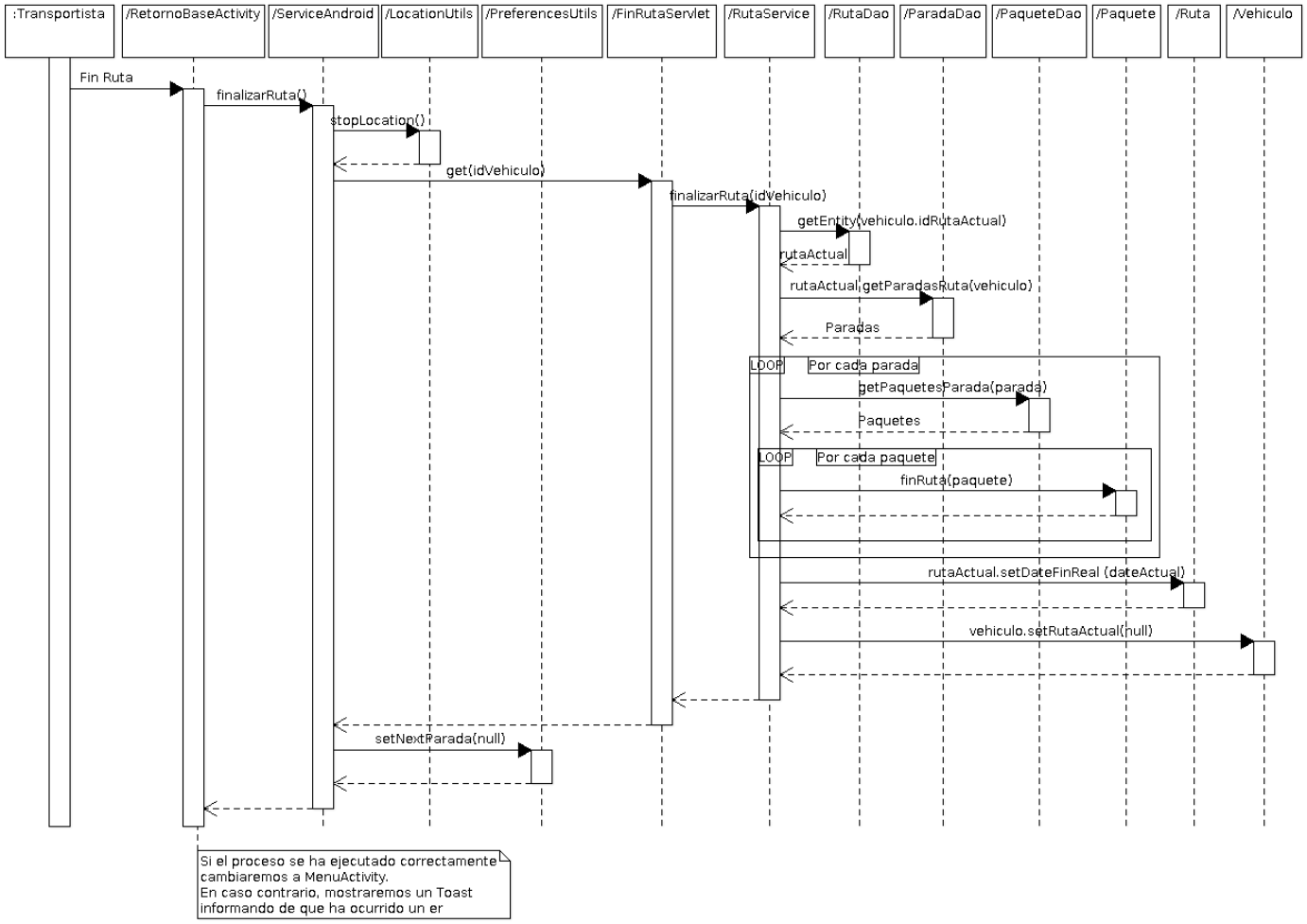


Fig. 18 Diagrama secuencia finalización ruta

3.3.10 Eliminar ruta

Desde el panel de control un administrador puede eliminar una ruta. En ese caso se ejecuta el siguiente proceso:

1. El administrador elimina la ruta.
2. El servidor comprueba que la ruta no se está ejecutando.
3. El servidor busca los paquetes asignados a la ruta y restablece sus estados.
4. El servidor elimina la ruta.
5. Si el proceso se ha ejecutado correctamente, mostrará la vista de la búsqueda de rutas.
6. En caso contrario, mostrará una alerta.

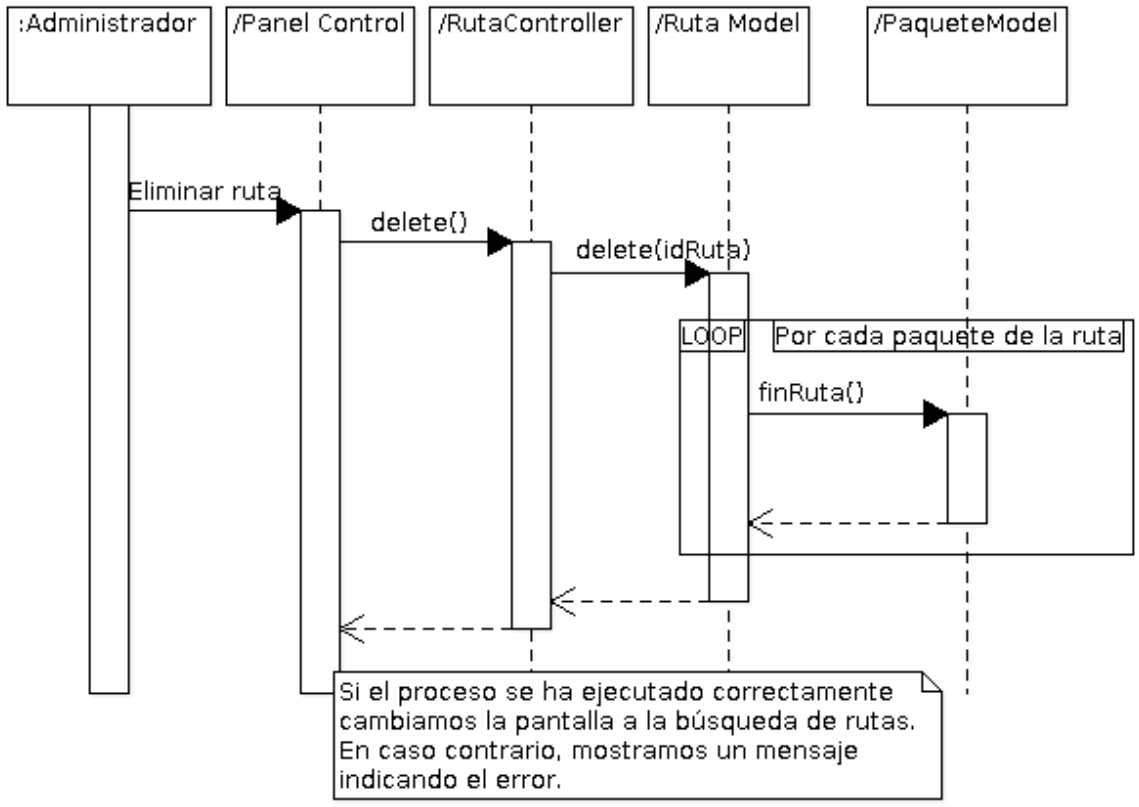


Fig. 19 Diagrama secuencia eliminar ruta

3.3.11 Eliminar vehículo

Desde el panel de control, un administrador puede eliminar un vehículo. En ese caso se ejecuta el siguiente proceso:

1. El administrador elimina el vehículo.
2. El servidor comprueba que el vehículo no esté realizando ninguna ruta.
3. El servidor busca las rutas que aún no se han realizado y están planificadas para este vehículo.
4. El servidor elimina dichas rutas siguiendo el proceso antes definido.
5. El servidor elimina el vehículo.
6. Si el proceso se ha ejecutado correctamente, mostrará la vista de la búsqueda de vehículos.
7. En caso contrario, mostrará una alerta.

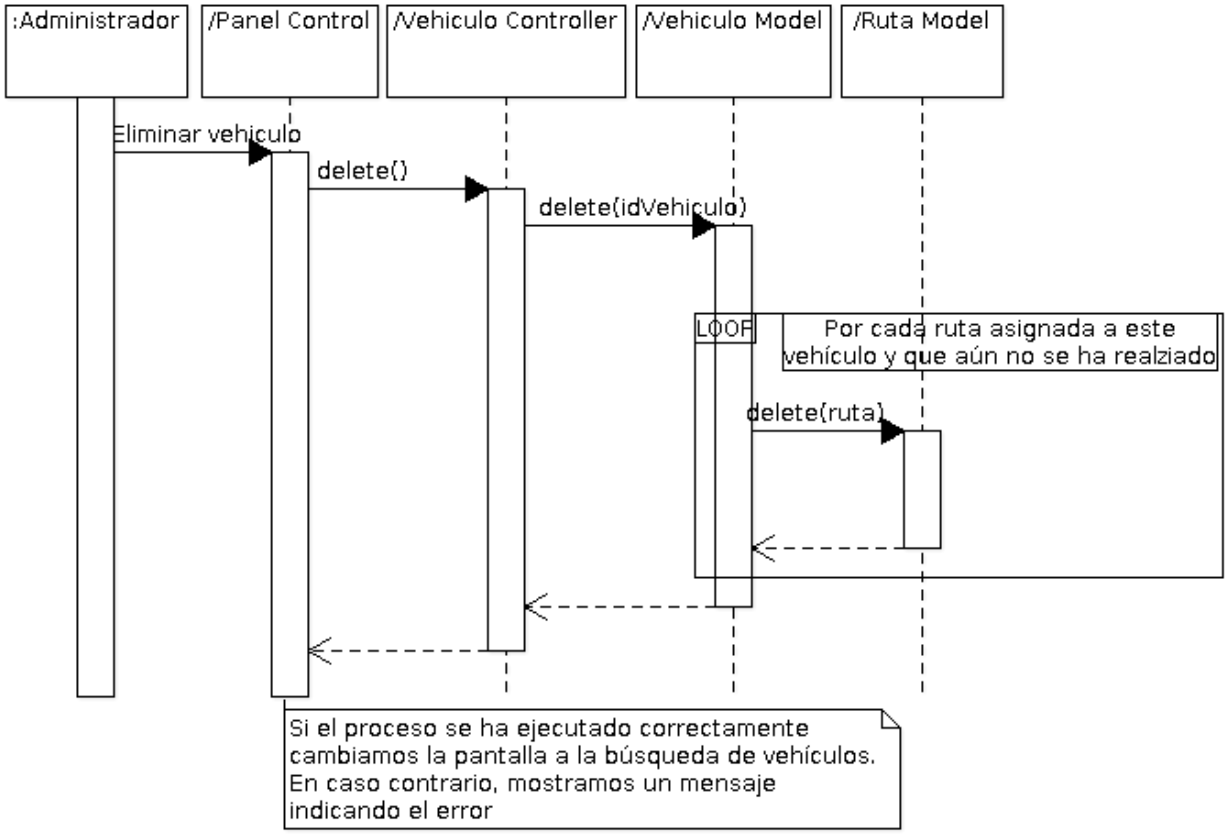


Fig. 20 Diagrama secuencia eliminar vehículo

3.3.12 Diagramas auxiliares

A continuación, se definen los diagramas de ejecución que complementan a los ya definidos. De forma que si dos procesos ejecutan la misma función, el diagrama de secuencia de esa función repetida se especifica en este apartado.

setIdVehiculo(idVehiculo)

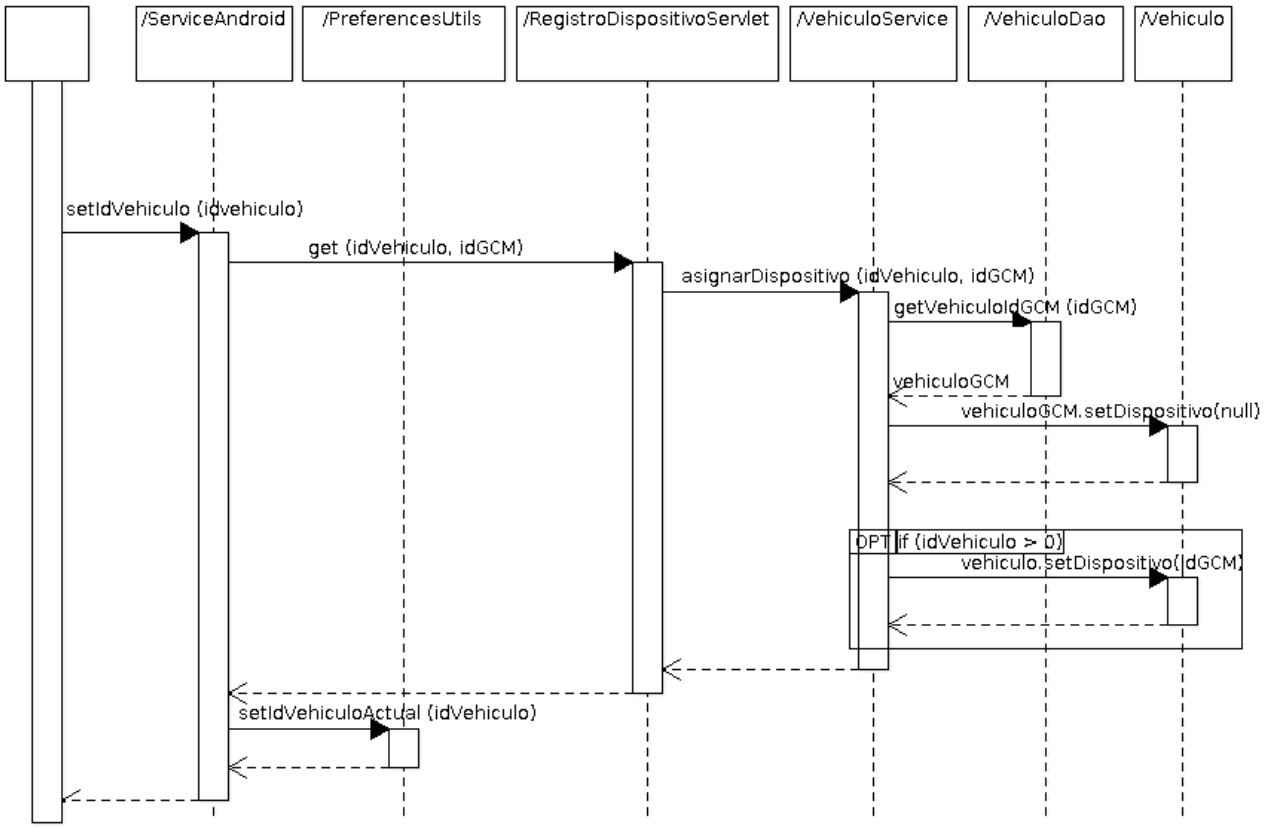


Fig. 21 Diagrama secuencia setIdVehiculo(idVehiculo)

updateNextParada()

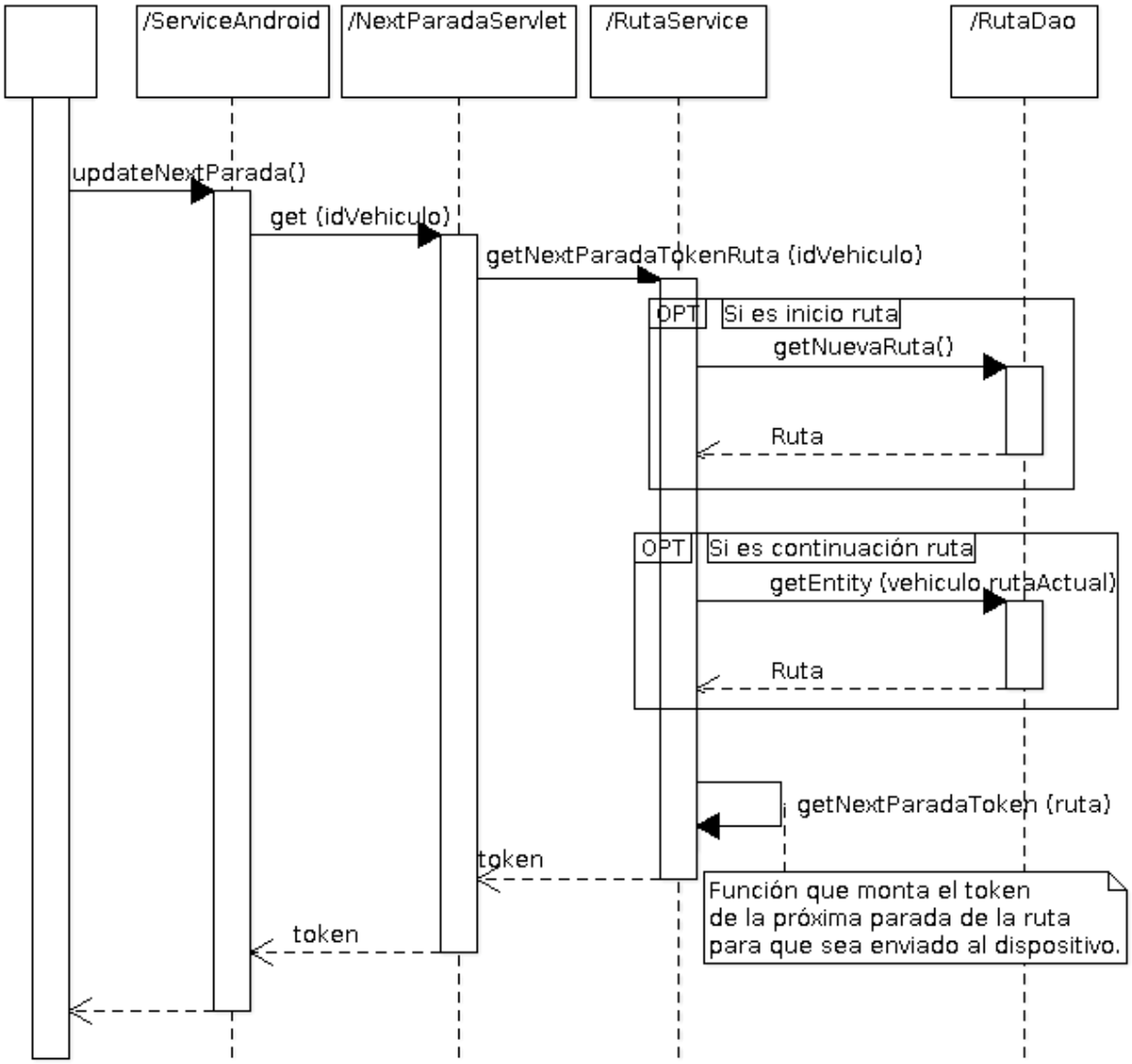


Fig. 22 Diagrama secuencia updateNextParada()

3.4 Diagrama de clases

A continuación, se define el diagrama de clases utilizado en el sistema Matroid:

- **Domain:** Esta capa contiene todo el dominio del sistema Matroid.
- **DAO:** DAO (*Data Acess Object)* nos proporciona una interfaz entre cada elemento del domain y su representación en base de datos. En su interior, la capa DAO utiliza un patrón *Factory* para reutilizar la conexión al servidor, aunque también podrá realizar una conexión nueva y personalizada siempre que se le indique.
- **Service:** Esta capa del proyecto contiene todas las funciones de negocio del sistema Matroid.
- **Android:** En esta capa está contenida toda la aplicación Android del sistema Matroid.

En este diagrama se pueden ver las relaciones entre las diferentes capas del sistema:

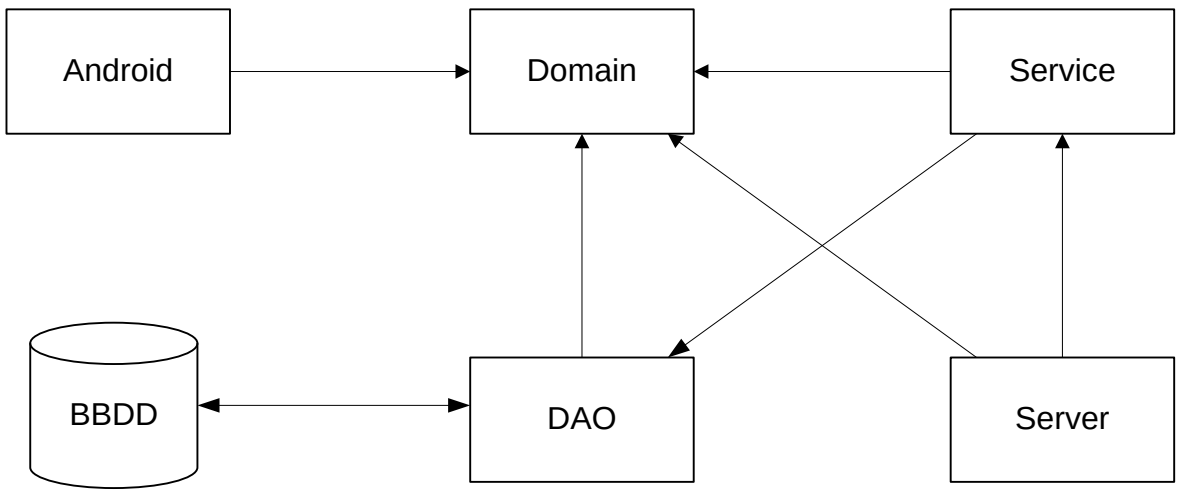


Fig. 23 Diagrama de clases

Para la capa web, que contiene el panel de control y la página de seguimiento, se ha utilizado la estructura que define el framework CodeIgniter para aprovechar todas sus capacidades al máximo.

3.5 Modelo entidad relación

En este apartado definiremos el modelo de entidad relación del sistema Matroid:

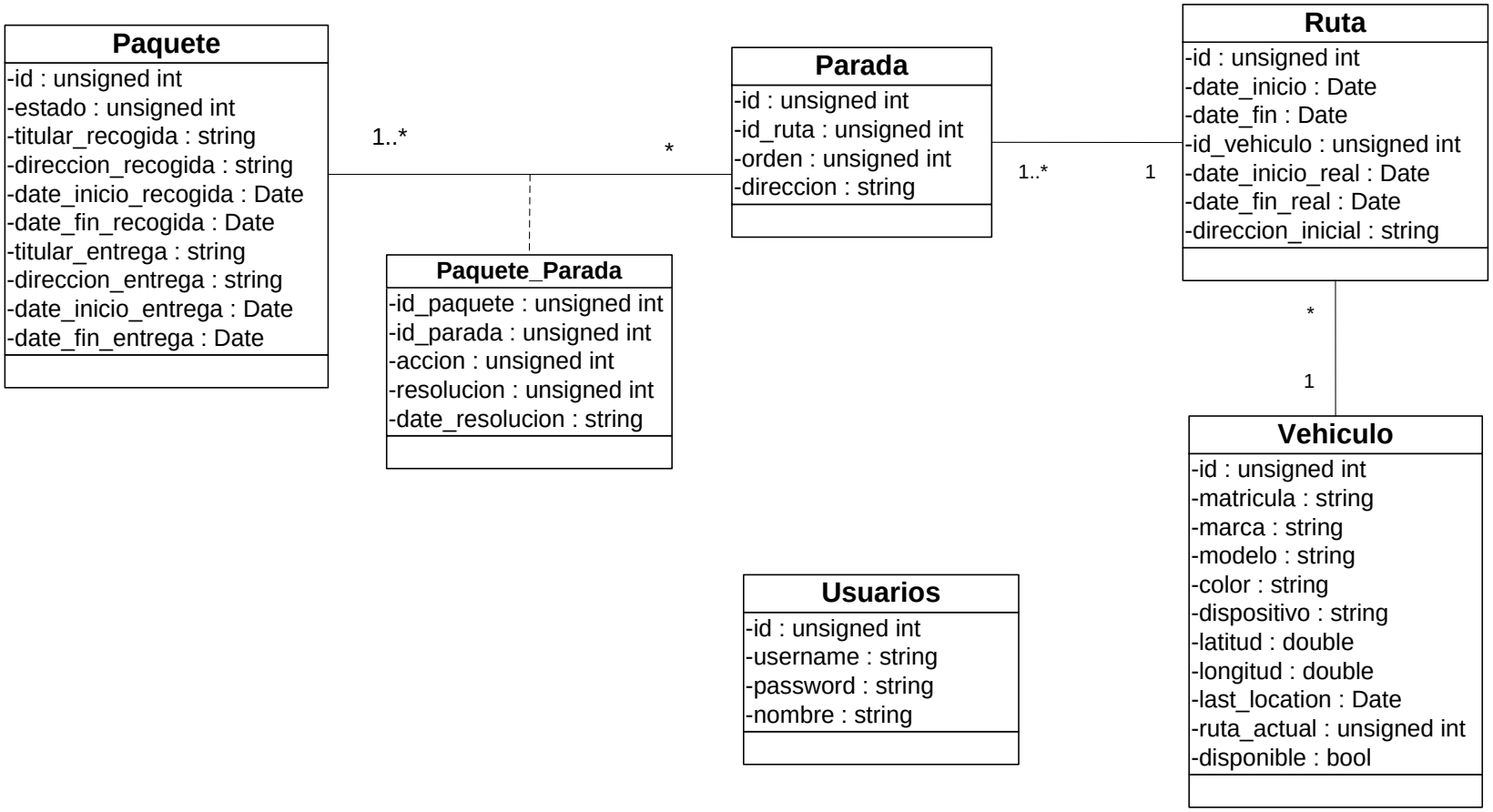


Fig. 24 Modelo entidad relación

3.6 Interfaz Android

La interfaz Android se divide en diferentes *activities* mediante las cuales el transportista podrá navegar e interactuar con la aplicación.

Cada *activity* de la aplicación está diseñada para mostrar la información necesaria según el momento concreto en el que se encuentre el transportista y las opciones de la actividad a realizar.

A continuación se definirán las *activities* que contiene la aplicación y el momento en el que se mostraran:

- Identificador GCM: *Activity* que se ejecutará al iniciar la aplicación y obtendrá el identificador GCM. Esta *activity* es transparente al transportista.
- Vincular vehículo: *Activity* que se mostrará al no tener un vehículo vinculado.
- Cambiar dominio server: *Activity* que se mostrará al cambiar el dominio del servidor al cual el dispositivo realiza las peticiones.
- Menú principal: *Activity* que se mostrará al tener un vehículo vinculado.
- LoadParada: *Activity* que comprueba si existen paradas por realizar en la ruta o hay que retornar a la base. Esta *activity* es transparente al transportista.
- Menú parada: *Activity* que se mostrará al dirigirse a la próxima parada de la ruta.
- Retorno base: *Activity* que se mostrará al haber completado todas las paradas de la ruta y se tenga que retornar a la base.
- Mapa: *Activity* que se mostrará para visualizar el mapa de la próxima parada o base.
- Navegación paso a paso: *Activity* que se mostrará para recibir las indicaciones de navegación hacia la próxima parada o base.

Seguidamente se mostrará un diagrama en el que se visualizan todas las *activities* y sus interacciones:

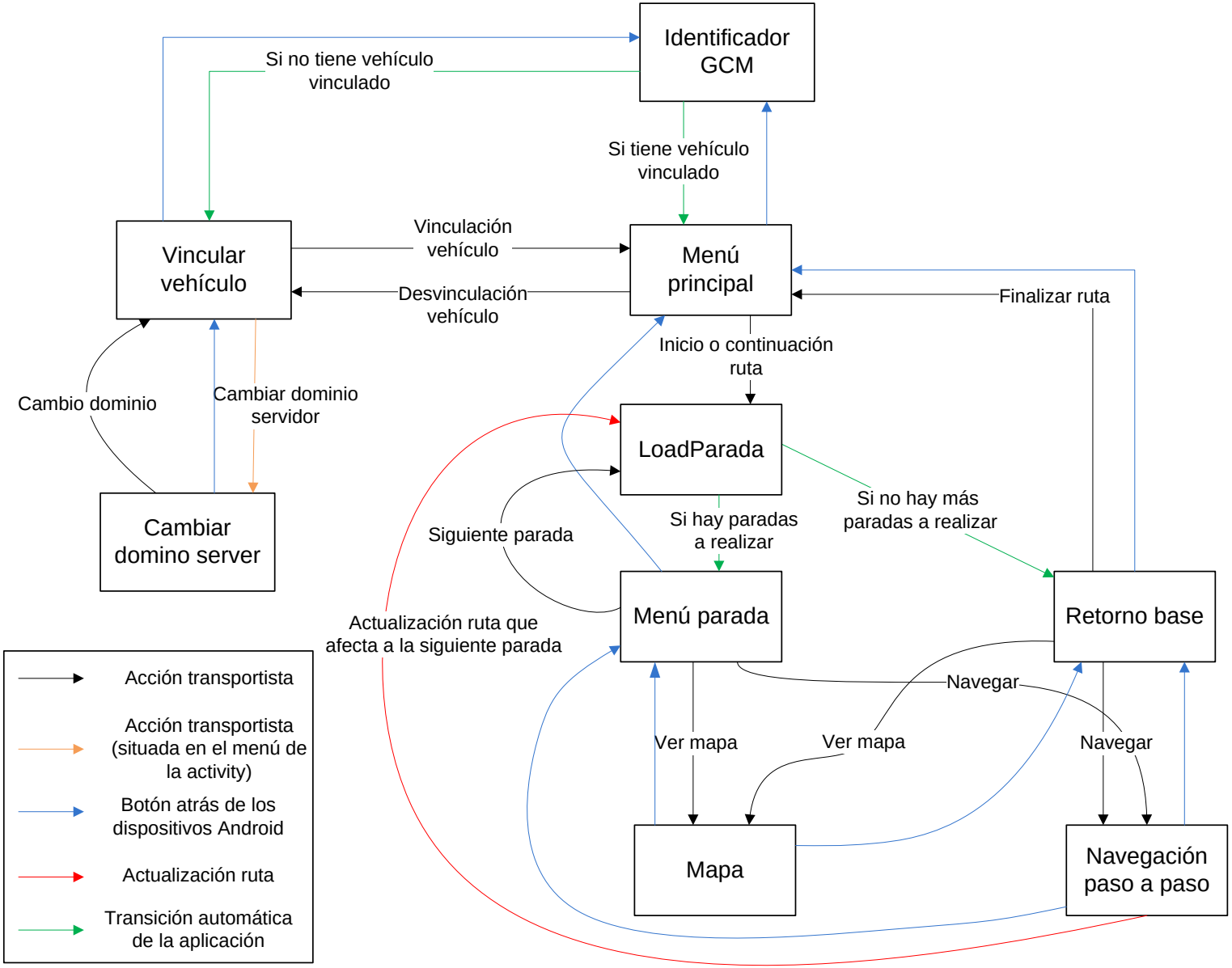


Fig. 25 Diagrama *activities* Android

3.7 Interfaz panel de control

En este apartado se definirá la estructura básica que tiene el panel de control.

El panel de control lo hemos dividido en cuatro secciones:

1. Header: Se trata de una barra horizontal, situada en la parte superior, que abarca todo el ancho de la pantalla. En esta barra encontramos la palabra *Matroid* que pulsándola nos dirigirá a la pantalla principal del panel de control. Además, en la parte derecha, encontramos el nombre y apellido del usuario que ha iniciado sesión. Pulsando sobre el nombre, se puede finalizar la sesión o editar el usuario.
2. Menú lateral: Se trata de un menú ubicado en la parte izquierda del panel de control y accesible desde cualquier pantalla. En este menú podremos cambiar a los tres grandes elementos del sistema Matroid: paquete, ruta y vehículo.
3. Container: Se trata de la parte principal del panel de control donde se realizan todas las funciones disponibles. Tiene un título para indicar qué función se está realizando en cada momento.
4. Footer: Se trata de la sección que indica la finalización del panel de control.

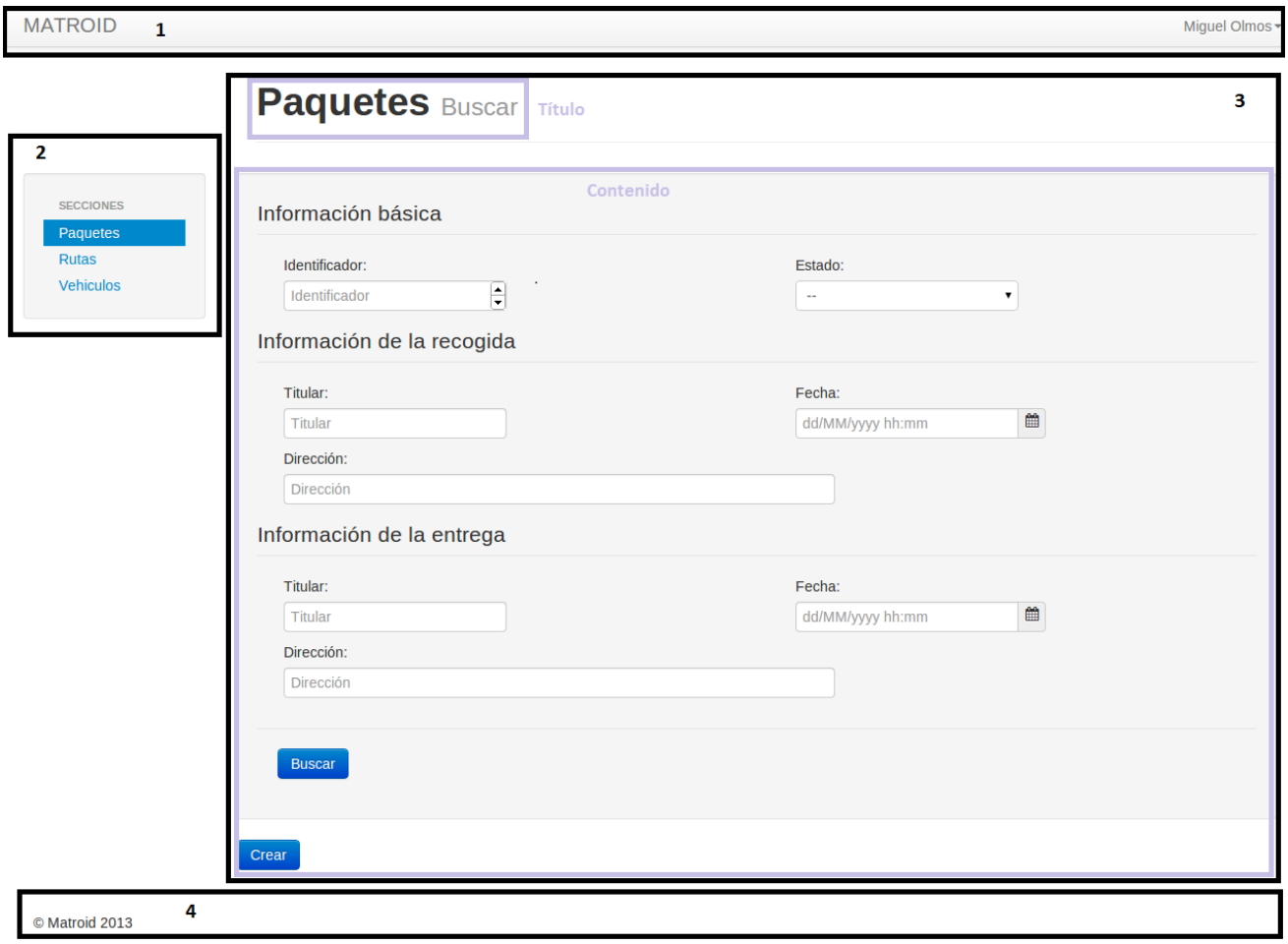


Fig. 26 Interfaz panel de control

3.8 Interfaz página de seguimiento

En este apartado se definirá la estructura básica que tiene la página de seguimiento.

La página de seguimiento la hemos dividido en cuatro secciones:

1. Header: Se trata de una barra horizontal, situada en la parte superior, que abarca todo el ancho de la pantalla. En esta barra encontramos la palabra *Matroid* que pulsándola nos refrescará la página de seguimiento.
2. Búsqueda: Se trata de la sección de la página de seguimiento que nos permite introducir el identificador de paquete que queremos buscar.
3. Resultado: Se trata de la sección que muestra todos los datos del paquete que se ha buscado.
4. Footer: Se trata de la sección que indica la finalización del panel de control.

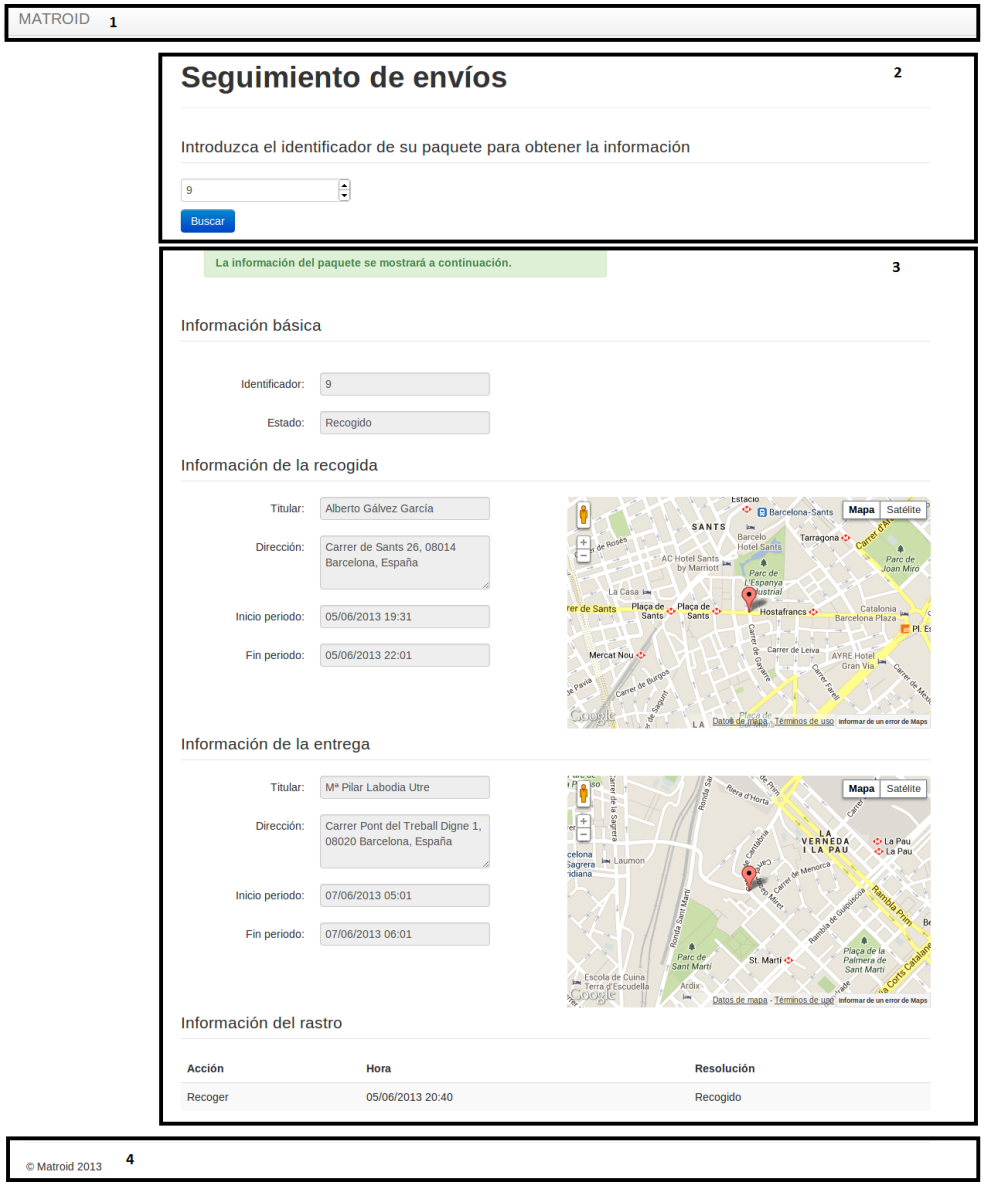


Fig. 27 Interfaz página de seguimiento

4 Detalles de la implementación

En este apartado se definirán y se explicarán los framework, las librerías, el gestor de base de datos y las tecnologías usadas en la implementación del sistema Matroid.

4.1 Tecnologías usadas

Las tecnologías más importantes del sistema que se han usado son:

4.1.1 Android

Android es un sistema operativo, basado en Linux, orientado principalmente a dispositivos móviles y tabletas con pantalla táctil, se puede encontrar en http://www.android.com/

Inicialmente el sistema operativo fue creado por la empresa Android Inc., pero en julio de 2006 Google la adquirió apoderándose del desarrollo del sistema operativo. Android tiene una cultura de software libre; por eso aunque sea un sistema operativo desarrollado por Google, la mayoría del código está liberado bajo la licencia Apache, que es una licencia libre y de código abierto.

Aunque Android tenga un núcleo basado en Linux y escrito en C, el lenguaje de programación de las aplicaciones es Java. Eso se debe a que Android ejecuta una máquina virtual llamada Dalvik que permite la compilación en tiempo de ejecución.

Por último, remarcar que gracias a su filosofía de código abierto y su máquina virtual Dalvik, Android no sólo está presente en dispositivos móviles y tabletas; sino que es posible ejecutarlo en infinidad de dispositivos y aparatos, como por ejemplo ordenadores, televisores, lavadoras,...

En este proyecto el sistema operativo Android es de vital importancia, ya que la aplicación que utilizan los transportistas está diseñada para ejecutarse en dispositivos móviles con este sistema operativo. Además, esto es clave por la gran variedad de dispositivos que tienen el sistema operativo Android, dando así un gran abanico de posibilidades a la empresa de paquetería que utilice el sistema Matroid.

Fig. 28 Logo Android

4.1.2 GPS

GPS (Global Positioning System) es un sistema de posicionamiento global de navegación por satélite que nos permite localizar a una persona, objeto o vehículo con una precisión de unos pocos metros en el uso estándar de esta tecnología. El sistema fue desarrollado, instalado y actualmente es operado por el Departamento de Defensa de los Estados Unidos.

El sistema se basa en una red de 24 satélites que orbitan sincronizadamente para cubrir toda la superficie de la Tierra. Para que una persona, objeto o vehículo pueda determinar su posición, es necesario un receptor que localice a 3 satélites como mínimo de forma que, calculando el tiempo que tardan las señales en llegarle y mediante triangulación, sea capaz de conocer la ubicación actual. Si existe la posibilidad de conectarse a más de 3 satélites, la precisión de la ubicación se verá aumentada.

La utilización del sistema GPS no conlleva ningún cargo económico, aunque los mapas cartográficos sobre los cuales se representa la información obtenida por el GPS sí suelen reportarlo.

En este proyecto, la utilización del sistema GPS es básica y necesaria para localizar la posición de todos los vehículos y también para la navegación giro a giro, que el transportista tiene la posibilidad de utilizar hacia su siguiente parada.

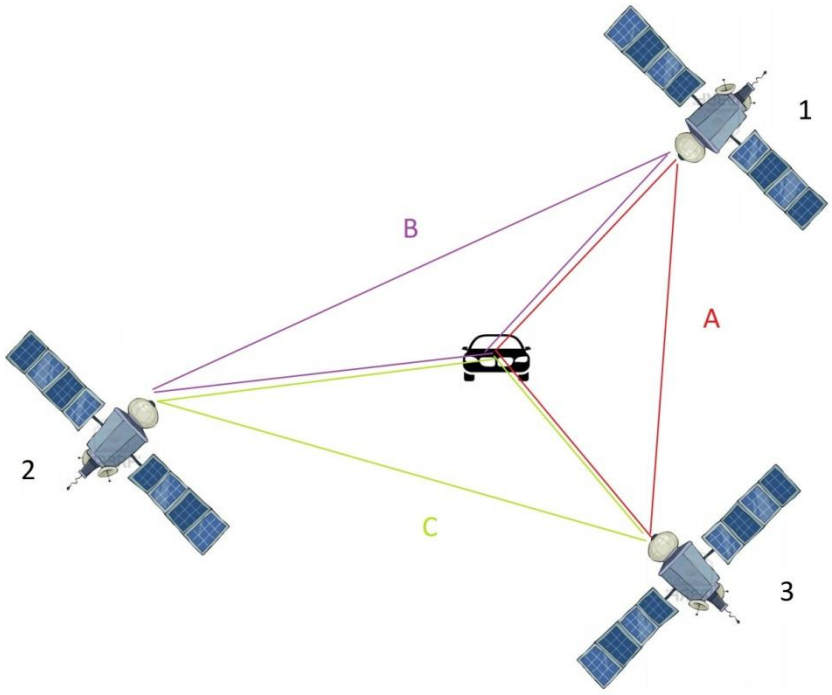


Fig. 29 Esquema GPS

4.1.3 Tecnología push

La tecnología push es un estilo de comunicaciones sobre Internet, en el cual es el servidor el que inicia la petición hacia el cliente.

Normalmente el cliente es el que inicia la petición al servidor y este último responde con la información solicitada. Aunque este sistema es el más usado y útil, en algunos casos determinados no es el que da mejores resultados.

Por ejemplo, si tenemos un cliente de email, es el cliente el que debe consultar cada cierto tiempo si hay un nuevo correo para él. Si está configurado para que realice la petición cada 15 minutos, podemos recibir el correo con un retraso de 15 minutos en el caso de que el correo hubiera llegado justo después de realizar una petición de consulta.

La solución a este problema es la tecnología push, ya que con ella no es el cliente el que tiene que realizar la consulta para saber si ha recibido un correo nuevo. Simplemente se deberá de registrar en el servidor y será este el que informará al cliente que ha recibido un nuevo correo, de esta manera se notifica de forma instantánea.

Esta tecnología es importante para nuestro sistema porque avisa al dispositivo móvil de manera inmediata, y de esta forma al transportista, de la modificación de la ruta actual. Además nos permite una reducción de gasto energético, al evitar que el dispositivo tenga que consultar cada cierto tiempo si ha habido una modificación en la ruta.

4.2 Lenguajes de programación

Los lenguajes de programación que se han utilizado para la implementación son los siguientes:

4.2.1 Java

Java es un lenguaje de programación de propósito general, concurrente, basado en clases y orientado a objetos, desarrollado originalmente por Sun Microsystems que fue adquirida por Oracle en abril de 2009. El lenguaje se encuentra bajo la licencia GNU GPL y se puede encontrar aquí http://www.java.com

La intención del lenguaje Java es permitir que los desarrolladores escriban el programa una sola vez y se ejecute en cualquier dispositivo sin que deba de ser recompilado, esto es conocido como WORA (*write once, run everywhere*). Para ello las aplicaciones Java deben de ejecutarse sobre una máquina virtual que ha de estar instalada en los dispositivos que queremos utilizar.

Dicha ejecución sobre una máquina virtual conlleva una reducción de rendimiento frente a los lenguajes nativos, desventaja que ha sido duramente criticada. No obstante, Java sigue siendo uno de los lenguajes más populares y usados del mundo.

En nuestro sistema Matroid, el lenguaje Java es utilizado para implementar tanto el servidor central como la aplicación Android.

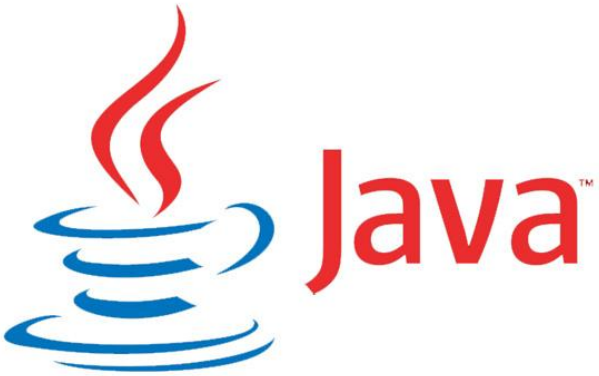


Fig. 30 Logo Java

4.2.2 PHP

PHP es un lenguaje de programación de propósito general y orientado a objetos, diseñado para la creación de páginas web dinámicas. El lenguaje se encuentra bajo la licencia PHP y se puede encontrar en http://php.net/

El código es interpretado por el servidor antes de enviarse al cliente. Cuando el cliente realiza una petición al servidor, este ejecuta el módulo procesador de PHP y devuelve el resultado para que sea enviado al cliente. Este resultado es dinámico dependiendo del programa PHP.

PHP puede ser ejecutado en la mayoría de sistemas operativos, ya sea Windows, Linux o Mac OS.

Sus principales características son:

- El código escrito en PHP es invisible desde la perspectiva del navegador y del cliente.
- Tiene capacidad de conexión con la mayoría de los motores de base de datos.
- Puede expandir su potencial utilizando módulos, llamados extensiones.
- Posee mucha flexibilidad que le otorga una gran acogida en muchas aplicaciones web.

Un inconveniente que se le achaca es que, al ser un lenguaje interpretado, su rendimiento se ve afectado comparado con lenguajes nativos.

En nuestro sistema Matroid el lenguaje PHP es utilizado para implementar el panel de control y la página de seguimiento.

Fig. 31 Logo PHP

4.2.3 SQL

SQL es un lenguaje declarativo, de acceso a base de datos relacionales que nos permite realizar diferentes operaciones sobre ella. Se puede encontrar en http://www.w3schools.com/sql/

El lenguaje se divide en dos conjuntos:

Un conjunto hace referencia al llamado Lenguaje de definición de datos o DDL, que es el que se encarga de la modificación de la estructura de los objetos de la base de datos. Existen cuatro operaciones básicas:

- CREATE: Permite la creación de un nuevo objeto.
- ALTER: Permite modificar la estructura de un objeto.
- DROP: Permite eliminar un objeto.
- TRUNCATE: Permite vaciar el contenido de una tabla.

El otro conjunto hace referencia al llamado Lenguaje de manipulación de datos o DML, que es el encargado de realizar tareas de consulta o manipulación de datos. Existen diversas operaciones, pero las más comunes son:

- SELECT: Permite consultar los datos de una o varias tablas.
- INSERT: Permite agregar uno o varios registros a una tabla.
- UPDATE: Permite modificar uno o varios registros de una tabla.
- DELETE: Permite eliminar uno o varios registros de una tabla.

En nuestro sistema Matroid, el lenguaje SQL es utilizado para realizar las operaciones contra la base de datos.

4.2.4 HTML 5

HTML5 es la última versión del lenguaje HTML, un lenguaje de marcado orientado a la elaboración de páginas web. Se puede encontrar en http://www.w3schools.com/html/html5_intro.asp

El lenguaje HTML ha evolucionado desde su origen en 1991, en la actualidad sigue evolucionando. En 1997, en su cuarta versión, el lenguaje HTML se publicó como recomendación del W3C (*World Wide Web Consortium*).

HTML5 aún se encuentra en versión experimental, pero ya ha sido utilizado por multitud de aplicaciones web por sus grandes ventajas. Algunas de ellas son:

- Nuevos elementos
- Nuevos atributos
- Soporte para CSS3
- Audio y vídeo
- Gráficos 2D/3D

En nuestro sistema Matroid, el lenguaje HTML5 es utilizado para la implementación del panel de control y la página de seguimiento.

Fig. 32 Logo HTML5

4.2.5 JavaScript

JavaScript es un lenguaje de programación interpretado, orientado a objetos y a la elaboración de páginas web. Se puede encontrar en http://www.w3schools.com/js/

En 1997 JavaScript fue adoptado como lenguaje estándar de la European Computer Manufacturers Association.

El código es interpretado desde el lado del cliente por el navegador, permitiéndole mejoras en la interfaz y añadiendo dinamismo a las páginas web. Esto se traduce en una mejor experiencia para el usuario que utiliza la web.

Todos los navegadores modernos interpretan el código JavaScript, para interactuar con la web utilizan el DOM (*Document Object Model*).

En nuestro sistema Matroid, el lenguaje JavaScript es utilizado para la implementación del panel de control y la página de seguimiento.

4.2.6 CSS

CSS, u hojas de estilo en cascada, es un lenguaje de hojas de estilos usado para maquetar la presentación de un documento escrito en lenguaje de marcas. Su aplicación más común es dar estilos a las páginas webs escritas en HTML y se puede encontrar en http://www.w3schools.com/css/

Tiene una sintaxis muy sencilla y fácil de usar. Una hoja de estilos consiste en una serie de reglas usando selectores.

Hay 3 formas de adjuntar la información de estilo a un documento HTML:

- Indicando con un atributo style en las etiquetas HTML el estilo de dicha etiqueta. Esta práctica no es muy recomendable, ya que no se encapsula la programación de la hoja de estilo y no es elegante a la hora de revisar el código. Su utilización se reserva para casos puntuales.

- Utilizando la etiqueta HTML <style>, consiste en implementar el código de maquetación dentro de la misma página HTML. Esta solución no es muy recomendable ya que no separa correctamente el código fuente del código de maquetación de la página web.

- Una hoja de estilo externa que consiste en desplazar todo el código de maquetación a un fichero externo y hacer una referencia desde la página HTML. Esta es la solución recomendada por W3C y es la que se debería de usar siempre que sea posible.

En nuestro sistema Matroid, el lenguaje CSS es utilizado para la maquetación del panel de control y la página de seguimiento.

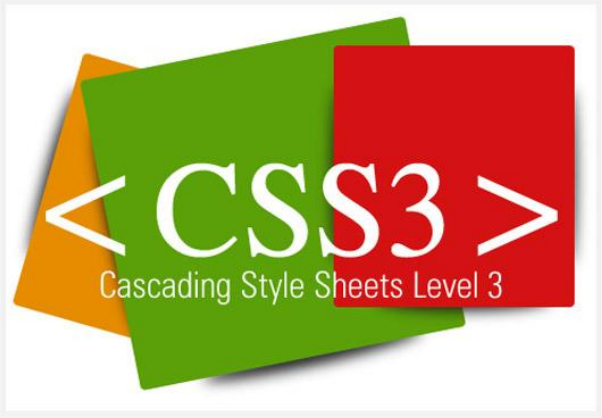


Fig. 33 CSS 3

4.3 Framework

Se han utilizado dos framework para implementar el panel de control y la página de seguimiento. A continuación se describirán y se especificarán las funciones principales que se han utilizado.

4.3.1 CodeIgniter

CodeIgniter es un framework para aplicaciones web basadas en PHP, de código abierto y caracterizado por su sencillez y rapidez. Se puede encontrar en: http://ellislab.com/codeigniter/user-guide/

Fig. 34 Logo CodeIgniter

Este framework ofrece una estructura sencilla y clara para realizar una página web siguiendo el patrón de 3 capas MVC. Además incorpora multitud de librerías que ayudan y facilitan la creación de páginas web y el mantenimiento de un código limpio.

Incluso Rasmus Lerdorf, el creador de PHP, expresó que le gustaba CodeIgniter “porque es rápido, ligero y parece poco un entorno”.

En nuestro sistema Matroid, el framework CodeIgniter se ha utilizado para la implementación del servidor web que contiene la página web de panel de control y la de seguimiento de paquetes.

A continuación se describirán los componentes que se han utilizado del framework CodeIgniter:

Las librerías que se han utilizado son:

- Session: Librería que nos permite gestionar las variables de sesión de forma más cómoda y sencilla.

- Form Validation: Librería que nos permite realizar una validación de forma prácticamente automática de los formularios web, además de proporcionarnos un extra de seguridad al realizar un filtro XSS para evitar la inyección de código JavaScript.

- Pagination: Librería que nos facilita la creación de una paginación a la hora de mostrar una gran cantidad de resultados. Esta librería es utilizada en todos los resultado de búsqueda de la aplicación.

- Active Record Class: Librería que nos permite generar consultas SQL mediante funciones. De esta forma la generación de las consultas es más fácil y segura, ya que aplica automáticamente capas de seguridad, como SQL Injection, a los parámetros de la operación.

Los helpers que se han utilizado son:

- URL Helper: Mediante este helper se pueden obtener datos de la URL, como por ejemplo parámetros, de forma sencilla. Este helper es utilizado, principalmente, para conocer todos los identificadores de elementos que informamos mediante la URL.

- Form Helper: Utilizando este helper es posible la creación de formularios con pocas líneas de códigos y muy clarificadoras. Convirtiendo así el código de un formulario, que puede ser grande y complicado de leer, en uno pequeño y mucho más legible. Este helper es utilizado siempre que es posible en la generación de formularios, aunque en algunos momentos no se podrá utilizar por las características del framework Bootstrap.

- Date Helper: Mediante este helper se pueden realizar operaciones y conversiones de formatos sobre fechas de manera sencilla. Este helper es utilizado para las conversiones de fechas a la hora de mostrarlas en la página web.

4.3.2 Bootstrap

Bootstrap es un framework de software libre destinado a la capa visual de aplicaciones web y está basado en JavaScript y CSS. Se puede encontrar en http://twitter.github.io/bootstrap/

Fig. 35 Bootstrap

El objetivo de este framework es facilitar el diseño y la ordenación de la capa visual, además de disponer de multitud de tipografías, formularios, botones, gráficos, barras de navegación..., que nos permite generar una página web rica en componentes visuales y con una experiencia del usuario más satisfactoria. Otra característica muy importante que nos ofrece este framework es la compatibilidad con los diferentes navegadores existentes. Ya sean en ordenadores, tablets o móviles, la aplicación web se adaptará automáticamente a las necesidades adaptando la capa visual de la aplicación.

La ordenación de la capa visual en este framework se llama Scaffolding y es uno de sus atractivos más importantes, ya que divide el contenido de todos los componentes visuales, incluida la página web en sí misma, en 12 columnas verticales; pudiendo definir claramente cuantas columnas debe de ocupar cada componente y evitar así tener que configurar el posicionamiento de cada uno de ellos.

En nuestro sistema Matroid, el framework Bootstrap se ha utilizado para la implementación de la capa visual de la página web del panel de control y de la de seguimiento de paquetes.

A continuación, se describirán los componentes más destacados que se han utilizado del framework Bootstrap:

- Navbar: Este componente nos permite dibujar una barra horizontal que realiza la función de cabecera de la aplicación web, además se puede fijar para que siempre sea visible. El componente ha sido utilizado para implementar la cabecera de la aplicación.

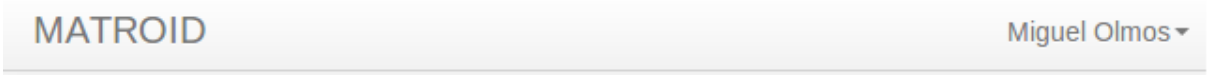


Fig. 36 Navbar

- Nav-list: Utilizando este componente dispones de un menú lateral sencillo pero con efectos visuales muy útiles. Este componente ha sido utilizado para implementar el menú lateral que contiene las diferentes secciones del panel de control.

Fig. 37 Nav-list

- Alerts: Este componente permite notificar de forma clara al usuario de la aplicación web sobre cualquier información, alerta o error que nos interese. En nuestra aplicación, el componente es usado siempre que se quiere mostrar un mensaje al usuario de la empresa de paquetería o al administrador del sistema Matroid.

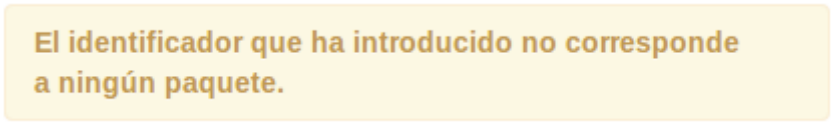


Fig. 38 Alert

- Modal: Este componente nos permite generar una ventana superpuesta al contenido principal, de esta forma el usuario de la aplicación web deberá atender al contenido de la modal antes de seguir con el contenido principal. Este componente ha sido utilizado para confirmar acciones importantes que puede realizar un administrador.

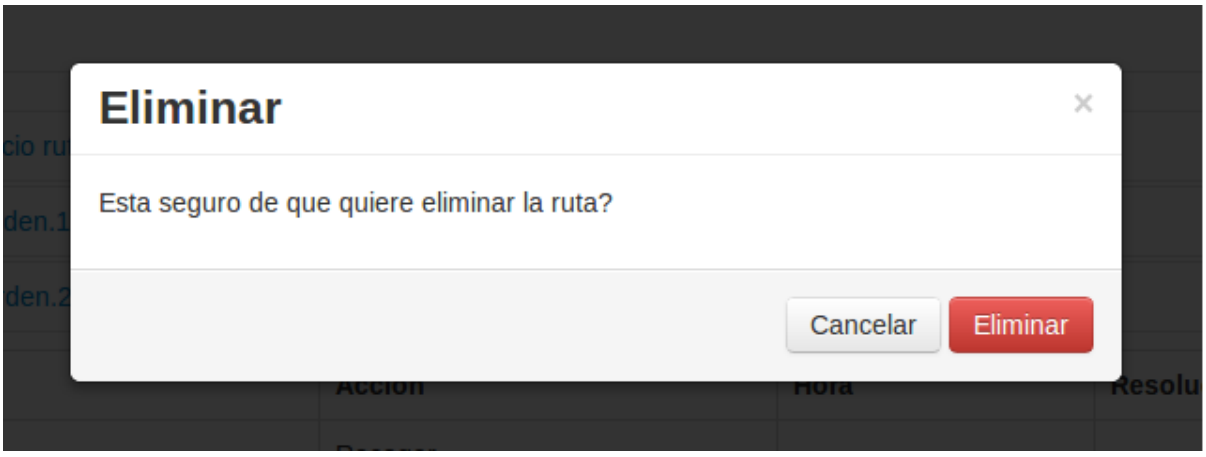


Fig. 39 Modal

- Dropdown: Mediante este componente se puede establecer un menú flotante al pulsar un elemento de la aplicación web que esté configurado correctamente. Este componente ha sido utilizado para mostrar las opciones de editar usuario y salir de la sesión cuando se pulsa sobre el nombre del usuario logueado.

Fig. 40 Dropdown

- Collapse: Utilizando este componente podemos compactar una gran cantidad de información que, si se mostrará toda de una vez, podría recargar mucho la página web y estropear la experiencia al usuario. Utilizando este componente la información queda oculta dejando únicamente visible un título, que especifica la información que contiene, hasta que el usuario clica sobre este, en ese momento la información se expandirá con un efecto visual. Este componente ha sido utilizado para colapsar la información de las diferentes paradas de una ruta.

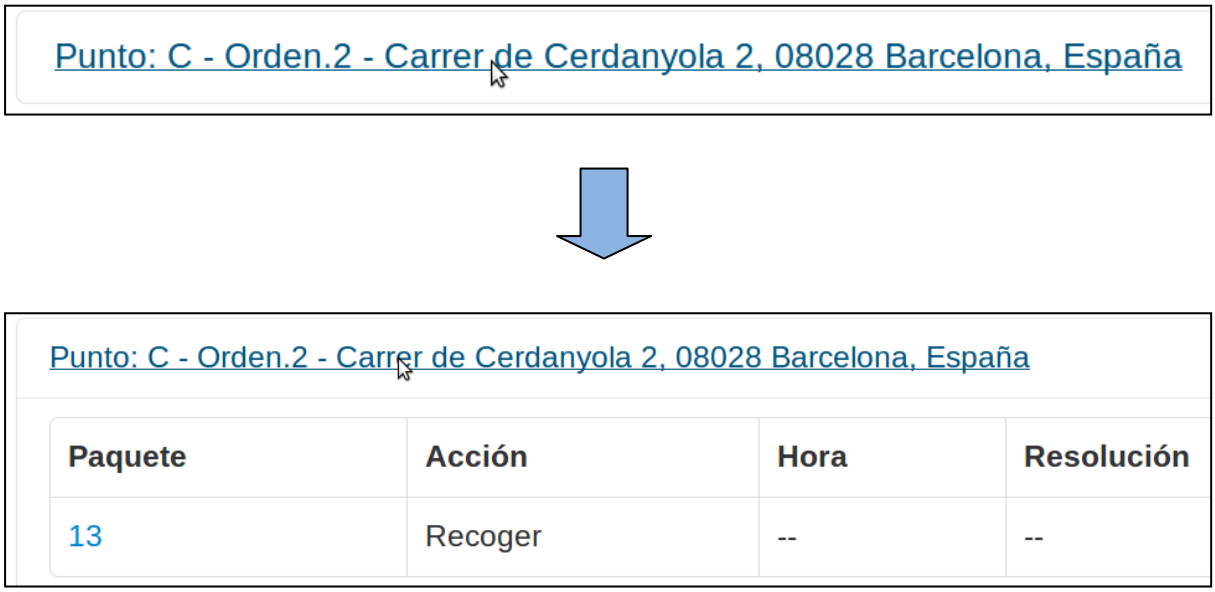


Fig. 41 Collapse

4.4 Gestor base de datos

Para almacenar toda la información necesaria se ha recurrido a un gestor de base de datos, muy utilizado por aplicaciones web, que es MySQL. Se puede encontrar en http://www.mysql.com/

MySQL es un gestor de base de datos relacional, multihilo y multiusuario, que desde abril de 2009 pertenece a Oracle Corporation. Su utilización requiere una licencia privativa en el caso que su utilización sea para un producto privado, mientras que se ofrece bajo la licencia GNU GPL para cualquier uso compatible con ella.

En nuestro sistema Matroid se ha decidido utilizar este gestor de base de datos por su facilidad de conexión, su gran velocidad y rendimiento, además de ofrecer una probabilidad muy reducida de corromper los datos.

Fig. 42 Logo MySQL

4.5 Servidores web

En el sistema Matroid se han utilizado dos servidores con diferentes propósitos. A continuación serán explicados cada uno de ellos y sus propósitos en el sistema.

4.5.1 Apache

El servidor Apache es el servidor web HTTP de código abierto más utilizado del mundo. Se puede encontrar en http://www.apache.org/

El servidor se desarrolla dentro del proyecto HTTP Server de la Apache Software Foundation y se ofrece bajo la licencia Apache Licence.

Sus principales ventajas son:

- Modular
- Código abierto
- Multi-plataforma entre Linux, Windows, Mac OS y otros.
- Popular, al ser el más usado se pueden encontrar soluciones a los errores fácilmente.

En nuestro sistema Matroid, este servidor es usado para alojar las aplicaciones webs de seguimiento de paquetes y panel de control.

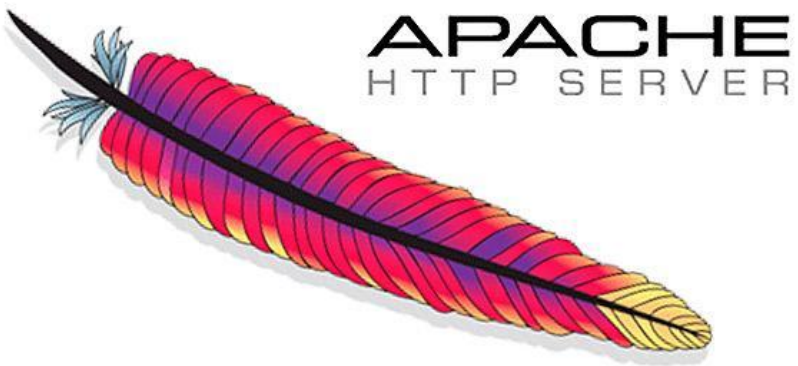


Fig. 43 Logo Apache

4.5.2 Tomcat

El servidor Tomcat es un servidor web, escrito en Java, con soporte de servlets y JSPs. Se puede encontrar en http://tomcat.apache.org/

El servidor se desarrolla bajo el proyecto Jakarta en la Apache Software Foundation y se ofrece bajo la licencia Apache License.

Una de sus ventajas es que, al ser escrito en Java, debe de ser ejecutado en una máquina virtual, lo que le permite ser utilizado en cualquier sistema operativo que disponga de la máquina virtual Java. Además es usado en entornos que requieren un alto nivel de disponibilidad y reciben un alto nivel de tráfico.

En nuestro sistema Matroid, este servidor es usado para recibir las peticiones de los dispositivos móviles mediante servlets y ejecutarlas.

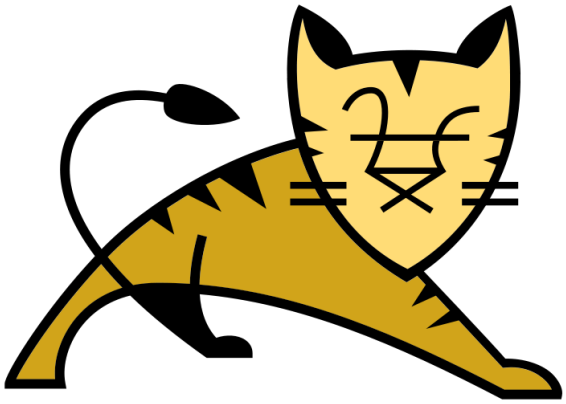


Fig. 44 Logo Tomcat

4.6 Bibliotecas/Librerías externas

En la implementación del sistema Matroid se han utilizado algunas bibliotecas y librerías externas que serán nombradas a continuación.

4.6.1 JQuery

JQuery es una biblioteca JavaScript que permite simplificar y escribir de forma más clara los códigos JavaScript, se puede encontrar en http://jquery.com/

JQuery es de software libre y de código abierto, posee la Licencia MIT y la Licencia Pública General de GNU v2, que se puede usar en todo tipo de proyectos.

Sus características principales son:

- Simplificar la manera de interactuar con los documentos HTML.
- Simplificar la manera de manipular el árbol DOM.
- Manipular hojas de estilos CSS.
- AJAX (*Asynchronous JavaScript And XML*).
- Efectos y animaciones.
- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.

En nuestro sistema Matroid su utilización se ve forzada por el framework Bootstrap, además de utilizarse para manipular algunos datos en el panel de control.

Fig. 45 Logo jQuery

4.6.2 Bootstrap-datetimepicker

Bootstrap-datetimepicker es una librería utilizada para mostrar un calendario en el momento de seleccionar las fechas, haciendo más fácil la tarea. Se puede encontrar en http://tarruda.github.io/bootstrap-datetimepicker/

Es una librería muy sencilla que permite múltiples opciones como definir el día inicial de la semana, activar o desactivar la selección de segundos en una hora, establecer fecha mínima y máxima...

Para utilizar esta librería se requiere el uso de Bootstrap y JQuery.

En nuestro sistema Matroid, la librería Bootstrap-datetimepicker ha sido utilizada en todos los campos de fechas de los formularios del panel de control.

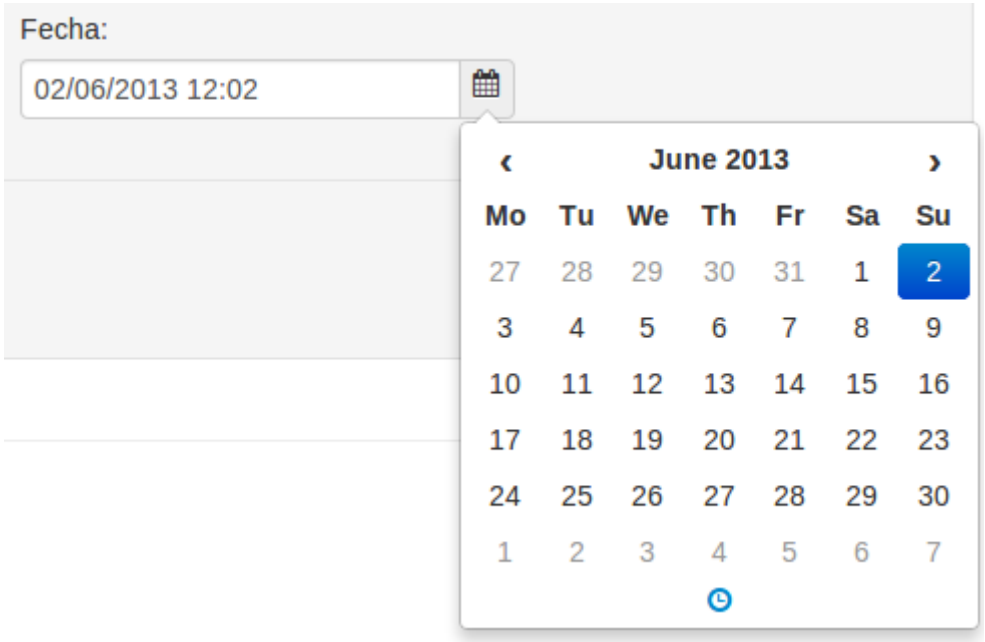


Fig. 46 Bootstrap-datetimepicker

4.6.3 GSON

Gson es una biblioteca de código abierto que nos permite manipular objetos JSON en Java, se puede encontrar en https://code.google.com/p/google-gson/

La función principal de esta biblioteca es la serialización y deserialización entre objetos Java y su representación en JSON, con una utilización de la librería muy sencilla.

Otras características son:

- Permite la conversión de objetos inmutables ya existentes.
- Tiene soporte para tipos genéricos de Java.
- Permite la representación personalizada de objetos.

Gson fue desarrollada por Google, de ahí el cambio de la 'G' de Gson por la 'J' de JSON, y más tarde fuera liberada bajo la licencia Apache License.

En nuestro sistema Matroid la utilización de esta biblioteca es básica, ya que todos los objetos que son transmitidos desde el servidor hacia los dispositivos móviles están formateados con JSON y se utiliza para serializar y deserializar estos objetos.

4.6.4 Log4j

Log4j es una biblioteca de código abierto que permite a los desarrolladores escoger la salida y el nivel de prioridad de los logs de un programa, se puede encontrar en http://logging.apache.org/log4j/1.2/

Por defecto log4j tiene 6 nieves de prioridad:

- FATAL: Se utiliza para escribir mensajes críticos. Normalmente, después de un mensaje de esta prioridad el programa abortará.
- ERROR: Se utiliza para escribir algún error ocurrido en el programa que no evita que se detenga.
- WARN: Se utiliza para escribir mensajes de alerta que queremos registrar pero que no afectan al funcionamiento del programa.
- INFO: Se utiliza para escribir mensajes que describen, desde un alto nivel, el comportamiento del programa.
- DEBUG: Se utiliza para escribir mensajes de depuración útiles para los desarrolladores cuando implementan un programa.
- TRACE: Su utilización es la misma que el DEBUG pero con un nivel de detalle mayor.

Log4j es un proyecto de la Apache Software Foundation y está ofrecido bajo la licencia Licencia Apache.

En nuestro sistema Matroid, la biblioteca Log4j es utilizada para registrar todos los logs del servidor central.

4.7 APIs externas

4.7.1 GCM

GCM (*Google Cloud Messaging*) es el servicio de Google mediante el cual podemos realizar peticiones push a dispositivos Android, sin tener un control sobre que IP tiene en ese momento el dispositivo (ya que son dinámicas), ni si está apagado o no, o si tiene cobertura o no. Se puede encontrar en http://developer.android.com/google/gcm/index.html

Fig. 47 Logo GCM

El servicio GCM se compone de tres elementos:

- La aplicación Android que recibe las notificaciones.
- Servidor propio que será el encargado de lanzar las notificaciones.
- Servidores GCM que son los servidores de Google que actúan como intermediarios.

Hay que tener en cuenta que, siempre que se habla del servicio GCM, el concepto notificación no está relacionado obligatoriamente a las notificaciones del sistema operativo Android, simplemente es un mensaje que se envía al dispositivo móvil y este reaccionará en consecuencia.

Para utilizar el servicio de forma correcta deberemos de realizar los siguientes pasos:

1. La aplicación Android se registrará en los servidores GCM y obtendrá un identificador único (registrationId).
2. La aplicación deberá de enviar el registrationId al servidor propio para que este lo almacene.
3. Cuando el servidor propio quiera enviar una notificación a la aplicación, este la deberá enviar a los servidores GCM.
4. El servidor GCM enviará la notificación, cuando sea posible, a la aplicación.

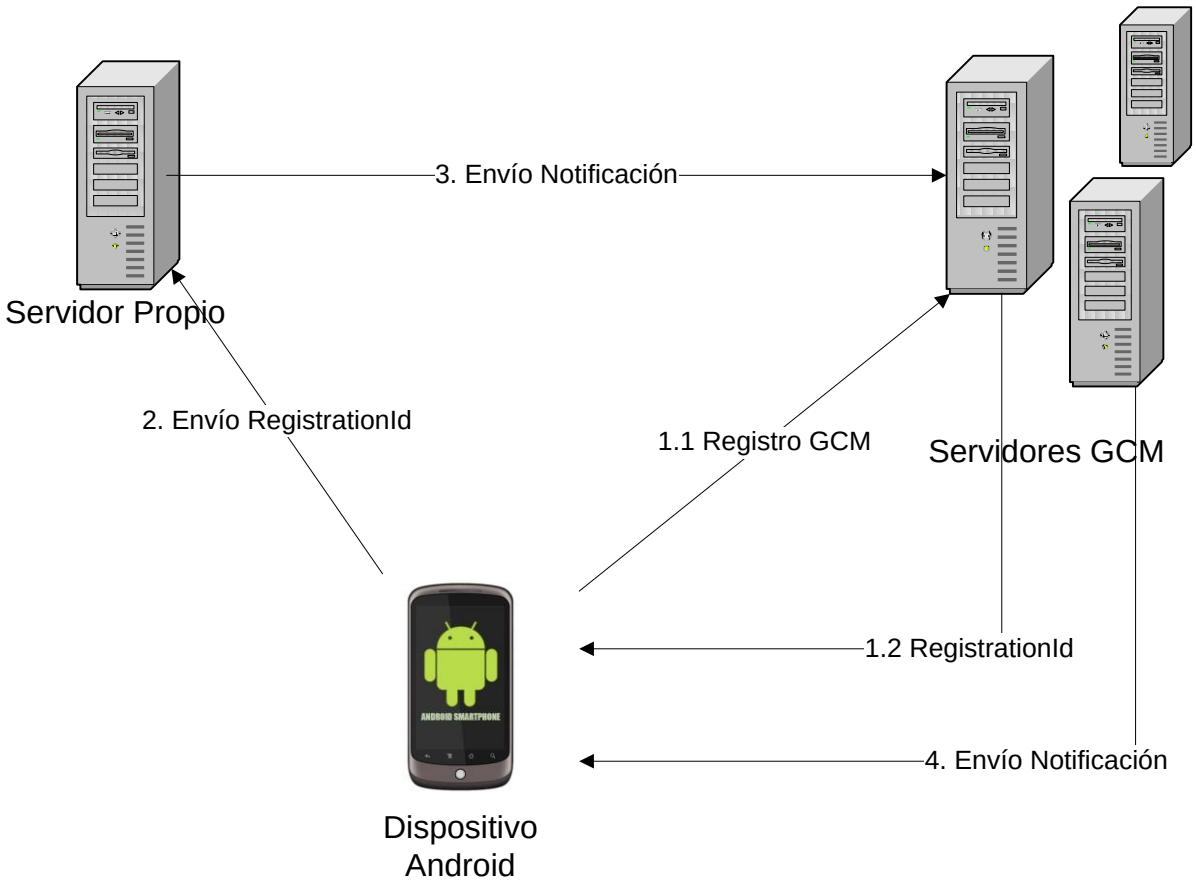


Fig. 48 Esquema GCM

Además este servicio obliga a:

- Disponer de una cuenta gratuita en los servicios de Google APIs.
- Disponer de las librerías necesarias para manipular el API GCM.

Y tiene:

- Un límite de 100 notificaciones sin ser entregadas.
- Un límite de tamaño de las notificaciones de hasta 4KB.

En nuestro sistema Matroid, el servicio GCM es utilizado para notificar a los dispositivos cuando se ha modifica la ruta y les afecta a la parada que se están dirigiendo.

4.7.2 Google Maps API

Google Maps API es un servicio de Google que nos permite insertar mapas de Google Maps en aplicaciones web y dispositivos Android.

En nuestro sistema Matroid se han utilizado dos versiones de la API:

Google Maps API v3

Esta API permite insertar Google Maps en aplicaciones web. Concretamente requiere de un cuyo contenido será rellenado y manipulado mediante JavaScript. Dentro de este div tendremos un mapa con todas las capacidades posibles: zoom, desplazamiento, vista satélite...

Dispone de una limitación de uso gratuita que alcanza las 25000 peticiones diarias, para disponer de más peticiones será necesario realizar un pago.

Google Maps Android v2

Esta API permite insertar Google Maps en aplicaciones Android y requiere de un FragmentActivty donde se insertará el mapa. El mapa tendrá todas las capacidades disponibles de la aplicación Android oficial de Google Maps.

Ambas APIs son configurables para concretar ubicación, limitar opciones o añadir marcadores, para ser utilizadas se necesita una cuenta gratuita en los servicios de Google APIs.

En nuestro sistema Matroid, esta API ha sido utilizada para representar los mapas en la aplicación Android, en el panel de control y en la página de seguimiento.

Fig. 49 Logo Google Maps

4.7.3 API de matriz de distancia

La API de matriz de distancia es un servicio web de Google, al que se accede mediante HTTP o HTTPS, que nos permite calcular la distancia y el tiempo aproximado entre dos o diversos puntos.

Para utilizar la API es necesario informarle de diferentes parámetros, los más relevantes son:

- Origen: Uno o más orígenes que pueden ser informados mediante la dirección o sus coordenadas.
- Destino: Uno o más destinos que pueden ser informados mediante la dirección o sus coordenadas.
- Mode: Especifica el modo de transporte con el que se realizará la ruta, acepta los valores: driving, walking, bicycling.

También es necesario saber que esta API dispone de unas limitaciones de uso gratuito que alcanzan los 100 elementos por consulta, 100 elementos cada 10 segundos y 2500 elementos cada 24 horas. Para disponer de más capacidad será necesario realizar un pago.

En nuestro sistema Matroid, se ha utilizado esta API para calcular la distancia y el tiempo aproximado entre paquetes, vehículos-paquetes y vehículos-base.

4.7.4 API de codificación geográfica

La API de codificación geográfica es un servicio web de Google, al que se accede mediante HTTP o HTTPS, que nos permite calcular una dirección dadas las coordenadas o las coordenadas dada una dirección.

La limitación de uso de esta API de forma gratuita alcanza las 2500 solicitudes al día. Para disponer de más capacidad será necesario realizar un pago.

En nuestro sistema Matroid, esta API ha sido utilizada para establecer la dirección aproximada de un vehículo.

5 Pruebas

En este apartado se especificaran las pruebas que se han realizado al sistema Matroid para comprobar el funcionamiento de este y llevarlo a situaciones límite, analizando su respuesta.

5.1 Aplicación Android

En las pruebas unitarias de la aplicación, aparte de realizar las pruebas simples para comprobar el funcionamiento, nos hemos centrado en que la aplicación funcione correctamente en el mayor número de dispositivos posibles. Estos son los dispositivos en los cuales hemos probado la aplicación y su resultado:

- HTC Wildfire (Android 2.2): La aplicación se ejecutaba algo lenta al realizar cualquier acción comparada con otros dispositivos. Además, no se podía visualizar los mapas en la aplicación ya que no tiene soporte para OpenGL ES 2.
- Samsung ACE (Android 2.3.5): La aplicación se ejecutaba mejor que en el caso del HTC, pero se notaba una falta de potencia por parte del dispositivo. Los mapas se visualizaban correctamente.
- Samsung Galaxy S I (Android 2.2.1): Es el dispositivo con especificaciones más bajas en el que la aplicación se ha ejecutado de forma fluida y sin ningún problema.
- Samsung Galaxy S II (Android 4.0.3): La aplicación se ejecutaba de forma correcta y se podían utilizar todas las funciones.
- LG Optimus 5 (Android 4.0.3): La aplicación se ejecutaba de forma correcta aunque dio un error al abrir el mapa, concretamente al definir la ubicación, y como consecuencia se mostraba un mapa con un zoom muy alto y sin marcador.
- Galaxy Tab 2 10.1 (Android 4.0.3): La aplicación se ejecutaba de forma correcta y se podían utilizar todas las funciones.
- Nexus 4 (Android 4.2.2): La aplicación se ejecutaba de forma correcta y se podían utilizar todas las funciones.

Además se realizaron pruebas para comprobar que la desactivación de la red o el GPS por parte del usuario, no implicara el quedarse con la aplicación bloqueada ni ejecutar ningún “forzar cierre”.

5.2 Servidor central

En las pruebas del servidor central se comprobó que todos los procesos se ejecutan correctamente, devolviendo el resultado esperado y realizándose sin errores las ejecuciones demandadas. Además las pruebas se centralizaron en el comportamiento del servidor en situaciones anómalas en las que se puede encontrar. Estas pruebas son las siguientes:

- Desconexión de base de datos.
- Desconexión del web service externo.
- Resultado erróneo del web service externo.
- Desconexión de la red.
- Error al ejecutar las APIs externas.
- Parada e inicio del servidor.
- Parámetros erróneos por parte de los dispositivos.
- Intento de realización de una acción por parte de un dispositivo no vinculado al vehículo que se corresponde.

En todas estas situaciones el servidor ha realizado el comportamiento previsto que es el siguiente:

- Continuar el proceso siempre que sea posible, si por ejemplo el error sólo afecta a una parte del proceso.
- Si no es posible continuar el proceso, finalizarlo y realizar rollback para no corromper los datos.
- En caso de un error producido por la acción de un dispositivo, el servidor le informará del error.
- En caso de que un dispositivo intente realizar una acción que no le corresponde, se le denegará el acceso.

5.3 Panel de control y página de seguimiento

En las pruebas de las aplicaciones web, panel de control y página de seguimiento, se ha comprobado el compartimento de todos los procesos que se pueden realizar en los diferentes navegadores. Los navegadores que se han comprobado son:

- Chrome 27.0.1453
- Firefox 20.0
- Internet Explorer 10
- Chrome for Android 27.0.1453

En todos los navegadores probados las aplicaciones web funcionaron de forma correcta y sin ningún problema al realizar cualquier acción. Cabe remarcar, que los elementos HTML se visualizan de forma diferente dependiendo del navegador. En algunos navegadores ofrecerá alguna función adicional, como es el caso del Chrome en los inputs numéricos que da la posibilidad de aumentar o disminuir el identificador del paquete mediante flechas.

Una limitación encontrada en la API de Google Maps, que se utiliza para visualizar los mapas, fue la no disponibilidad de visualizar rutas con más de 8 hitos sin contar el origen ni el destino, aumentando este límite hasta 23 en el caso de Google Maps for Business. Este límite nos afecta en la visualización del detalle de una ruta, ya que en el caso de que tenga más de 8 paradas el mapa no se podrá visualizar.

También se realizó una prueba de las aplicaciones web restringiendo la ejecución del código JavaScript. En esta prueba los elementos se visualizaban correctamente, pero algunas acciones no se podían realizar. Estas acciones son:

- No se visualizan los mapas.
- No se puede utilizar el calendario a la hora de introducir las fechas.
- No se pueden eliminar elementos (paquetes y vehículos).
- No se pueden desvincular vehículos.
- No se puede mostrar el cálculo de la dirección aproximada de un vehículo.
- No se mantendrán los datos de los formularios en según que situaciones.

5.4 Pruebas de estrés

Para intentar poner el sistema Matroid al límite, se aplicaron tres pruebas de estrés al sistema para comprobar cómo reaccionaba.

Para las pruebas se utilizó un PC con las siguientes características:

- Procesador: CPU Intel DUO 2 E6750 2.67GHz
- Memoria RAM: 2GB DDR2
- Sistema Operativo: Ubuntu 10.04

En el caso de la red, las características son:

- Velocidad de bajada: 10Mb
- Velocidad de subida: 820 Kbps

A continuación, se detallarán las pruebas que se han realizado y sus resultados:

Página de seguimiento

La prueba de estrés en la página de seguimiento es importante, ya que es la página que estaría abierta al público y por lo tanto recibiría muchas peticiones.

Las características de la prueba que se ha realizado son las siguientes:

- Carga de la página con un resultado de búsqueda de paquete.
- Tiempo de la prueba 10 minutos.
- Los usuarios se han ido incrementando hasta llegar a los 100 usuarios concurrentes.

El resultado de la prueba lo podemos ver en el siguiente gráfico:

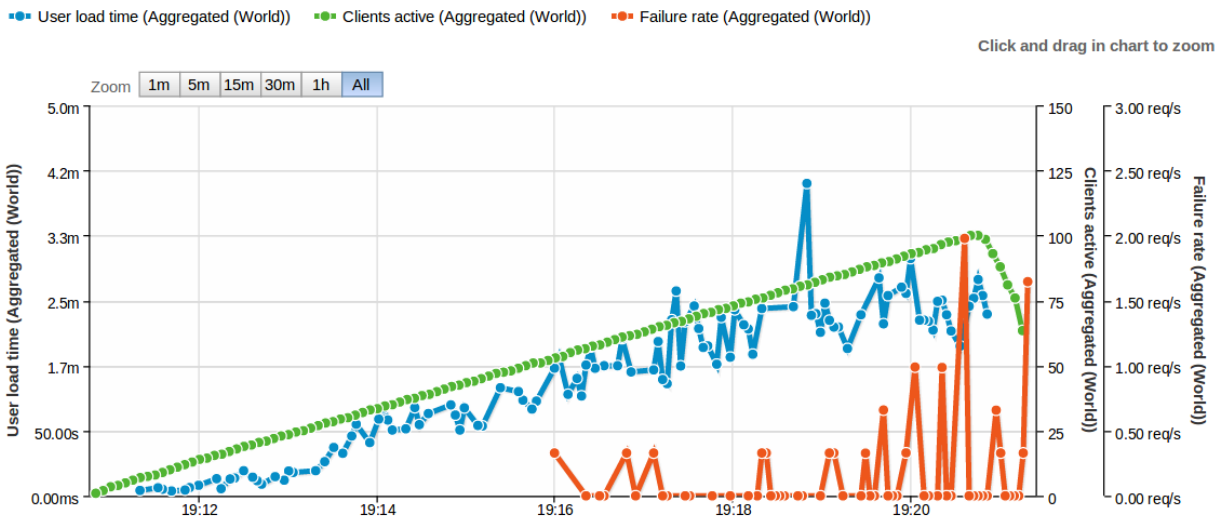


Fig. 50 Gráfico prueba página de seguimiento

Las conclusiones que se obtienen son las siguientes:

- La experiencia para el usuario se verá afectada si en ese momento existen más de 25 usuarios concurrentes, ya que el tiempo de carga se incrementa hasta 50 segundos.
- Al llegar a los 50 usuarios concurrentes, la espera en la página web es extremadamente larga, alcanzando casi los 2 minutos, haciéndola inaccesible desde el punto de vista del usuario.
- Cuando se supera los 85 usuarios concurrentes la tasa de error en la transmisión de datos es preocupante, incrementándose cada vez más a medida que aumentan los usuarios.

Petición de siguiente parada

Se ha escogido hacer la prueba de petición de siguiente parada, ya que es una de las que más solicitarán los transportistas y además es la segunda de mayor complejidad a realizar.

Las características de la prueba que se ha realizado son las siguientes:

- Simular la petición de un dispositivo de la obtención de la siguiente parada.
- Tiempo de la prueba 10 minutos.
- Las peticiones se han ido incrementado hasta llegar las 100 peticiones concurrentes.

El resultado de la prueba lo podemos ver en el siguiente gráfico:

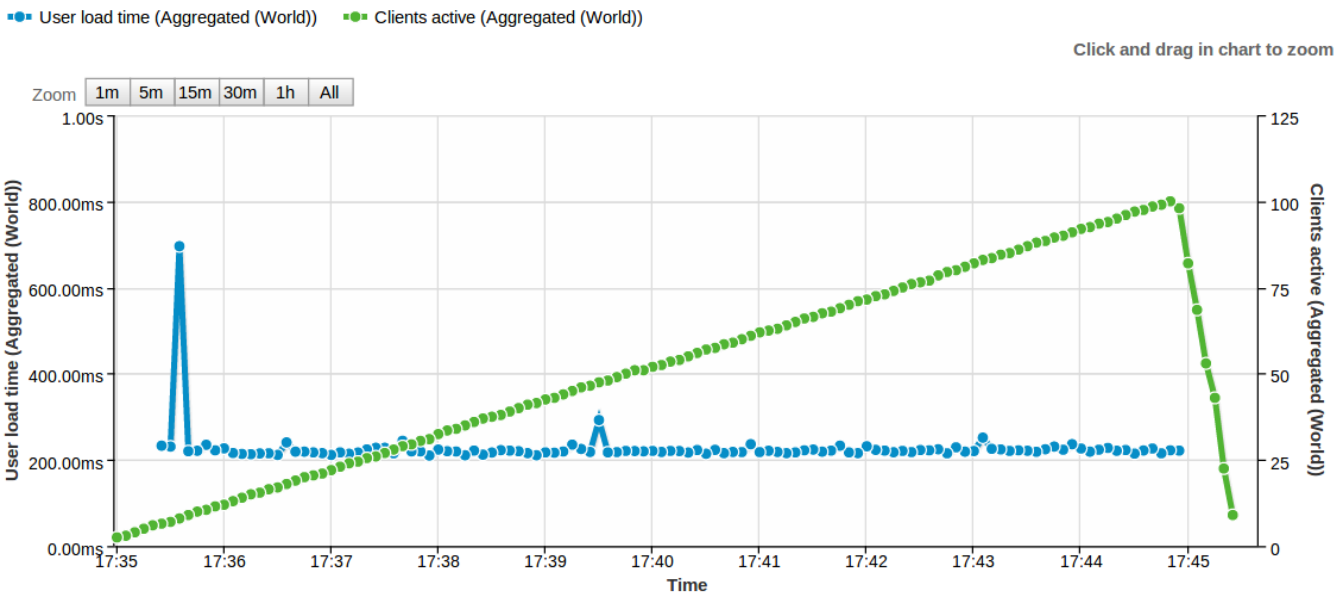


Fig. 51 Gráfico prueba petición siguiente parada

Las conclusiones que se obtienen son las siguientes:

- El tiempo de respuesta es de 200ms y es estable incluso llegando a las 100 peticiones concurrentes.
- Aunque a priori se podía pensar que la operación requeriría un esfuerzo por parte del servidor, vemos que este puede ejecutar los procesos sin problemas.
- El tiempo de respuesta tan pequeño también es debido a que sólo enviamos datos de pequeño tamaño, ya que únicamente informamos de la siguiente parada y no existe ningún elemento pesado como podría ser una imagen.

Actualización de la ruta

La prueba de la actualización de la ruta es una de las más importantes, ya que es la operación de más complejidad a realizar cuando la ruta se está ejecutando.

Las características de la prueba que se ha realizado son las siguientes:

- Se simulará la petición de actualización de la ubicación de un vehículo, lo que conllevará al cálculo de la actualización de la ruta.
- Se ha anulado el filtro que evita que un dispositivo sólo pueda ejecutar la operación de actualización cada 5 minutos.
- Tiempo de la prueba 10 minutos.
- Las peticiones se han ido incrementado hasta llegar a las 100 peticiones concurrentes.

El resultado de la prueba lo podemos ver en el siguiente gráfico:

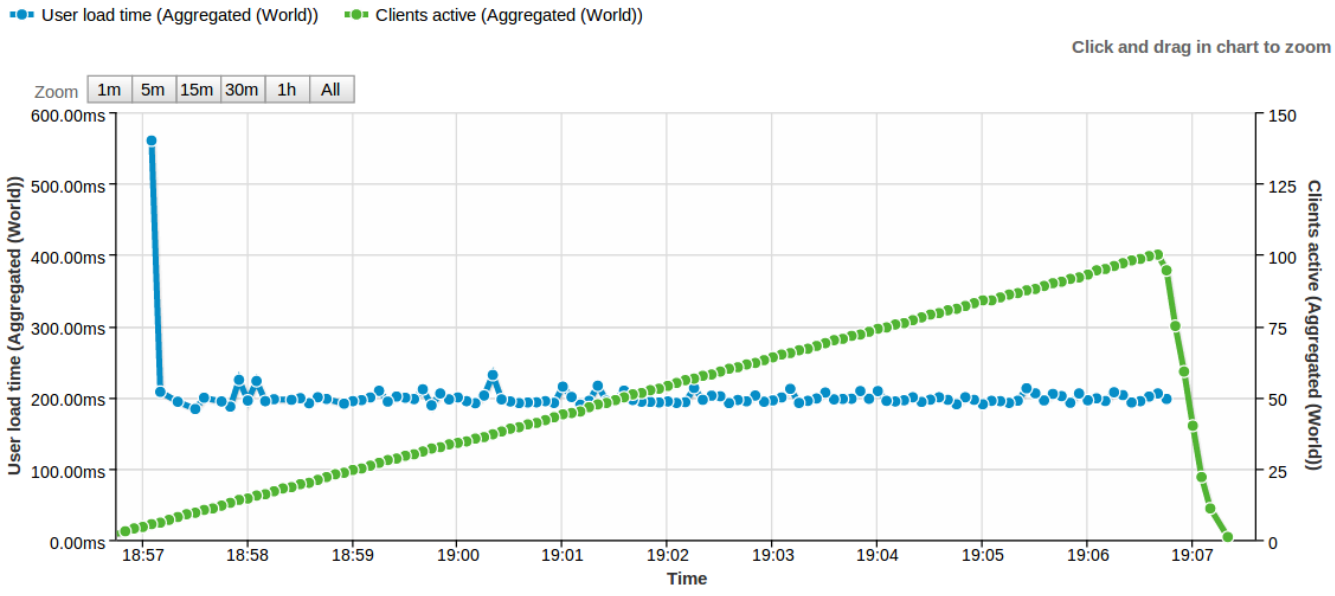


Fig. 52 Gráfico prueba actualización ruta

Las conclusiones que se obtienen son las siguientes:

- El tiempo de respuesta es de 200ms y es estable incluso llegando a las 100 peticiones concurrentes.
- Los resultados son buenos porque el proceso de actualización de la ubicación de un vehículo genera un nuevo thread para realizar el cálculo de actualización de la ruta, lo que nos permite un tiempo de respuesta muy corto.
- Aunque todo parezca buenos resultados, al realizar la prueba se detectó un problema. Si existen muchos paquetes disponibles para realizar el cálculo de actualización, este se puede demorar un tiempo prologando lo que provocará que los demás thread tarden más tiempo en ejecutarse o simplemente sean abortados por el sistema. Como consecuencia, es posible que una actualización llegue al dispositivo demasiado tarde o se pierda la oportunidad de recorrer una ruta más óptima. El problema se solventará aplicando la restricción de tiempo que ha sido anulada para la prueba.

6 Manual

6.1 Manual de instalación

6.1.1 Requerimientos básicos

Para poder ejecutar el sistema Matroid deberemos de disponer de los siguientes elementos:

- Dispositivo Android con versión 2.2 o superior
- Maquina virtual Java
- Servidor Tomcat 6
- Servidor apache2
- Modulo de PHP para el servidor apache2
- Gestor de base de datos MySQL

6.1.2 Aplicación Android

Para que la aplicación Android funcione correctamente necesitamos un dispositivo con Android 2.2 o superior, que tenga instalado la Play Store y con una cuenta de Gmail asociada al dispositivo.

Una vez que disponemos de un dispositivo con las características requeridas, para instalar la aplicación Android deberemos de realizar los siguientes pasos:

1. Introducir el APK en el dispositivo móvil.
2. En la configuración del dispositivo activar una opción para permitir instalar aplicaciones de “orígenes desconocidos”. En cada dispositivo es diferente, pero normalmente la opción está en: Ajustes -> Seguridad -> Orígenes desconocidos
3. Instalar el APK previamente introducido en el dispositivo.
4. Cuando la instalación haya finalizado, estará la aplicación lista para ser ejecutada.

6.1.3 Servidor central

Para instalar la aplicación central deberemos de realizar los siguientes pasos:

1. Crear la base de datos ejecutando el contenido del fichero *Matroid.sql*. Esta base de datos ya tendrá contenido de ejemplo.
2. Copiar la carpeta *MatroidFiles* en el *home* del usuario que ejecutará el Tomcat (típicamente será en /usr/share/tomcat6/). Dentro de esta carpeta nos encontraremos con los siguientes ficheros:
    a. *config.properties*: Fichero de configuración de la base de datos.

b. *calculoRutasIniciales.xml*: Fichero que contiene la respuesta de nuestro web service de prueba al ejecutar el proceso de creación de rutas.
c. *actualizacionRuta.xml*: Fichero que contiene la respuesta de nuestro web service de prueba al ejecutar el proceso de actualización de rutas. Este fichero se encontrará comentado y debe de seguir así hasta que no queramos actualizar una ruta.

En ambos ficheros XMLs se deberá de definir qué respuesta daría el web service de cálculo según la situación actual. Inicialmente estos ficheros contendrán un ejemplo para la primera ejecución, en el fichero *calculoRutasIniciales.xml* para la primera generación de rutas y en el *actualizacionRuta.xml* para la primera actualización (en este último caso se deberá de descomentar cuando realmente se quiera actualizar).

3. Se deberá desplegar el *MatroidServer.war* en el Tomcat instalado. Para ello nos dirigiremos al Tomcat Manager, situado en http://www.dominio.com:8080/manager/html/, y desplegar el WAR.
4. Una vez arrancado el WAR, ya podremos ejecutar nuestro primer cálculo inicial de rutas. Para ello ejecutaremos el siguiente comando: *java –jar Matroid.jar*
5. Si queremos que el proceso de cálculo inicial sea cada 8 horas, debemos añadir el siguiente cron: ** */8 * * * usuario java –jar Matroid.jar*
6. Una vez configurado el servidor central y calculadas las primeras rutas iniciales, podremos utilizar la aplicación Android siguiendo las instrucciones del manual de usuario.

6.1.4 Panel de control y página de seguimiento

Para instalar el panel de control y la página de seguimiento deberemos realizar los siguientes pasos:

1. Activar *mod_rewrite* del Apache. Para ello deberemos seguir las siguientes instrucciones:
   a. Ejecutamos: *sudo a2enmod rewrite*
   b. Editamos */etc/apache2/sites-enabled/000-default* modificando el campo *AllowOverride* del <*Directory /var/www/*>, de *None* a *All*
   c. Reiniciamos Apache ejecutando: *sudo /etc/init.d/apache2 restart*
2. Copiar la carpeta *MatroidWeb* a */var/www*
3. Una vez copiada, entramos en *MatroidWeb/application/config/database.php* y configuramos los parámetros de la base de datos, que son los mismos que en el servidor central.
4. Después entramos en *MatroidWeb/application/config/config.php* y modificamos, si es necesario, el parámetro *base_url*.
5. Una vez realizados estos pasos deberíamos tener acceso tanto al panel de control como a la página de seguimiento.
   a. Panel de control: http://dominio/MatroidWeb/login
   b. Página de seguimiento: http://dominio/MatroidWeb

6.2 Manual de usuario

A continuación se definirá un manual para cada tipo de usuario que utilice el sistema Matroid.

6.2.1 Transportista

Se especificará el manual que debe de seguir el transportista para ejecutar la aplicación Android en su dispositivo móvil. Se supondrá, como hemos explicado en el manual de instalación, que la aplicación ya la tiene instalada correctamente.

1. Abrir la aplicación

Para abrir la aplicación se deberá ir al menú de aplicaciones del dispositivo móvil y pulsar sobre la aplicación Matroid.

Fig. 53 Lanzador aplicación Matroid

Una vez abierta se mostrará la pantalla para vincular un vehículo (en caso de no tenerlo ya vinculado) o se mostrará el Menú Principal.

2. Configuración servidor

Para configurar el servidor, al cual se debe de conectar el dispositivo para realizar todas las operaciones, se debe pulsar el botón **menú** (o los tres puntos) en la pantalla de **Vinculación al vehículo** y después pulsar sobre la opción **Cambiar dominio servidor**. Una vez que se haya pulsado esta opción, se accederá a una pantalla donde se podrá cambiar el dominio del servidor y luego, una vez modificado correctamente, se deberá pulsar **Aceptar**.

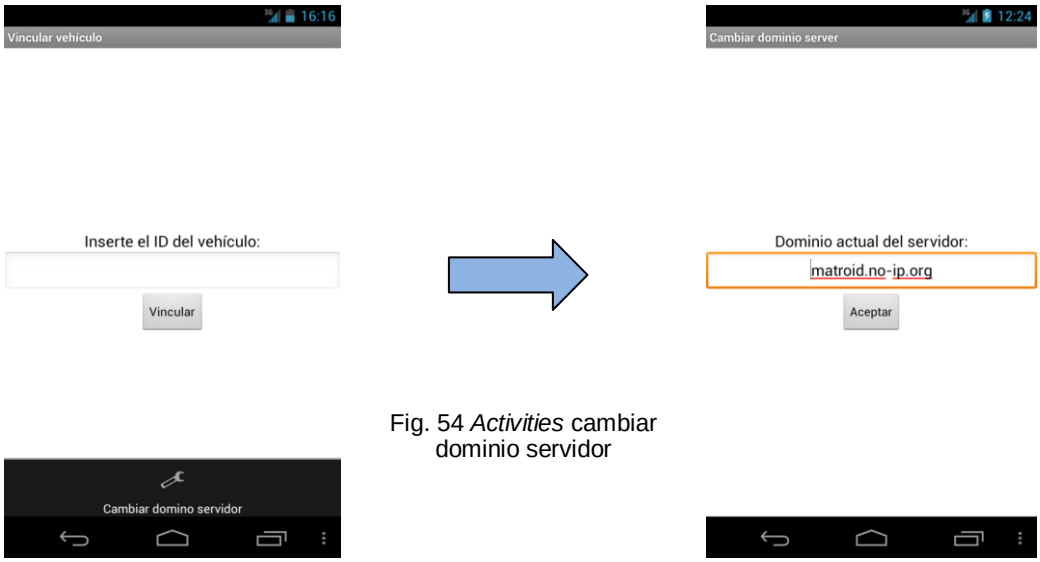


Fig. 54 *Activities* cambiar dominio servidor

3. Vinculación al vehículo

Para vincular el dispositivo móvil al vehículo se deberá de introducir el identificador del vehículo y pulsar **Vincular**. Si el dispositivo se ha vinculado correctamente, se accederá al **Menú Principal**.

En caso de error en la vinculación del vehículo o que el vehículo esté asignado a otro dispositivo, se mostrará una alerta.

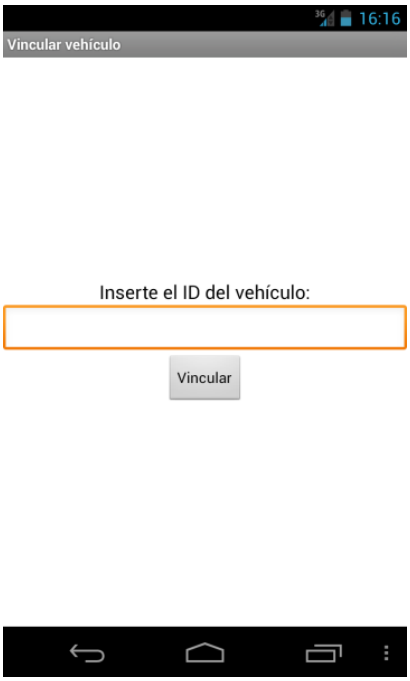


Fig. 55 *Activity* vincular vehículo

4. Menú Principal

El Menú Principal se mostrará una vez vinculado el dispositivo. En este menú se visualizará el identificador del vehículo que está vinculado y se podrá realizar las siguientes acciones:

1. Desvincular el dispositivo del vehículo.
2. Comenzar o retomar la ruta asignada.

Fig. 56 *Activity* menú principal

5. Desvinculación del vehículo

Para desvincular el vehículo se deberá pulsar el botón **Desvincular Vehículo** situado en el **Menú Principal**. Si la desvinculación se ha realizado correctamente, se accederá a la pantalla para vincular el dispositivo a un vehículo.

En caso de error en la desvinculación del vehículo, se mostrará una alerta.

6. Comenzar o retomar la ruta

Para comenzar o retomar la ruta que tiene asignada el vehículo, se deberá pulsar sobre el botón **Start Ruta** situado en el **Menú Principal**. Si existe una ruta en curso para el vehículo vinculado, se procederá a retomar dicha ruta; en caso contrario, se iniciará la ruta asignada al vehículo. Sea cual sea el caso, si la acción se ha ejecutado correctamente se accederá al **Menú Parada**.

En caso de que no haya ruta asignada para el vehículo vinculado se mostrará un mensaje de alerta.

7. Menú Parada

En el Menú Parada se visualizará la información sobre la siguiente parada de la ruta que se debe de realizar. Esta información es la siguiente:

a. Datos de la ruta que está realizando actualmente.
b. Datos de la próxima parada que debe realizar.
c. Datos de los paquetes que debe de entregar o recoger en la próxima parada a realizar.

Además, desde este menú se podrán realizar las siguientes acciones:

1. Visualizar en un mapa la ubicación de la siguiente parada a realizar, para ello se deberá de pulsar el botón **Ver mapa**.
2. Realizar una navegación giro a giro hasta la siguiente parada a realizar, para ello se deberá pulsar el botón **Navegar**.
3. Recoger o entregar los paquetes asignados a la parada, para ello se deberá pulsar el botón **Recogido** o **Entregado** del paquete, que se le mostrará según el caso, o el botón **No realizado**, si no se ha podido realizar la recogida o entrega.
4. Obtener los datos de la siguiente parada, para ello se deberá pulsar el botón **Siguiente Parada**. Para obtener los datos de la siguiente parada se deberá de haber informado sobre la recogida, entrega o la no realización de la acción, de todos los paquetes de la parada en curso; en caso de no ser así, se mostrarán los datos de la misma parada. En el caso de no haber ninguna parada más a realizar en la ruta, se mostrará el **Menú Retorno Base**.

En caso de haber algún error en alguna de las acciones, se mostrará un mensaje de alerta.

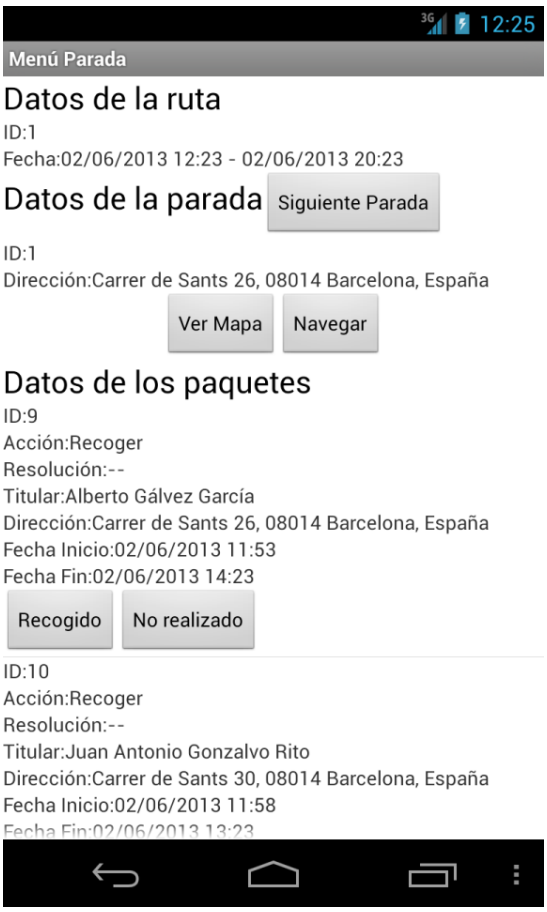


Fig. 57 *Activity* menú parada

8. Menú Retorno Base

Este menú se visualizará cuando no haya más paradas a realizar en la ruta actual. En él se mostrará la siguiente información:

a. Datos de la ruta que está realizando actualmente.
b. Datos de la base que debe retornar.

Además, desde este menú se podrán realizar las siguientes acciones:

1. Visualizar en un mapa la ubicación de la base, para ello se deberá de pulsar en el botón **Ver mapa**.
2. Realizar una navegación giro a giro hasta la base, para ello se deberá pulsar en el botón **Navegar**.
3. Finalizar la ruta actual, para ello se deberá de pulsar en el botón **Finalizar ruta**.

En caso de haber algún error en alguna de las acciones se mostrará un mensaje de alerta.

Fig. 58 *Activity* menú retorno base

9. Finalizar ruta

Una vez que se haya completado la ruta y se haya retornado a la base, se deberá de pulsar el botón **Finalizar ruta** situado en **Menú Retorno Base**. Si la acción se ha ejecutado correctamente se accederá al **Menú Principal**, donde se podrá volver a iniciar la siguiente ruta asignada al vehículo o desvincular el dispositivo del vehículo.

10. Actualización de la ruta

Siempre que se esté en modo navegación giro a giro hacia una parada o retornando a la base, si la actualización de la ruta afecta al destino al que se está dirigiendo se accederá automáticamente al **Menú Parada** con los datos actualizados, además de mostrar un mensaje avisando de que se trata de una actualización.

Fig. 59 *Activity* actualización ruta

11. Salir de la aplicación

Para salir de la aplicación Android se deberá pulsar sobre el botón *Home* del dispositivo.

6.2.2 Administrador

Se especificará el manual que debe de seguir el administrador para ejecutar el Panel de Control del sistema Matroid.

Se supone que el panel de control está situado en la misma máquina que utiliza el administrador para acceder a él, de forma que la dirección del servidor será *localhost*.

Siempre que sea posible consideraremos como un elemento una abstracción de paquete, ruta o vehículo con el fin de simplificar el manual.

1. Acceso

Para acceder al Panel de Control se deberá de introducir en un navegador web la siguiente dirección: http://localhost/MatroidWeb/login.html, así se accederá a la pantalla **Login Administrador**.

Una vez se ha accedido al **Login Administrador**, se deberá introducir el usuario y la contraseña. Si son correctos, se accederá a la pantalla principal del Panel de Control.

Si el usuario y/o contraseña son incorrectos se mostrará una alerta en pantalla.

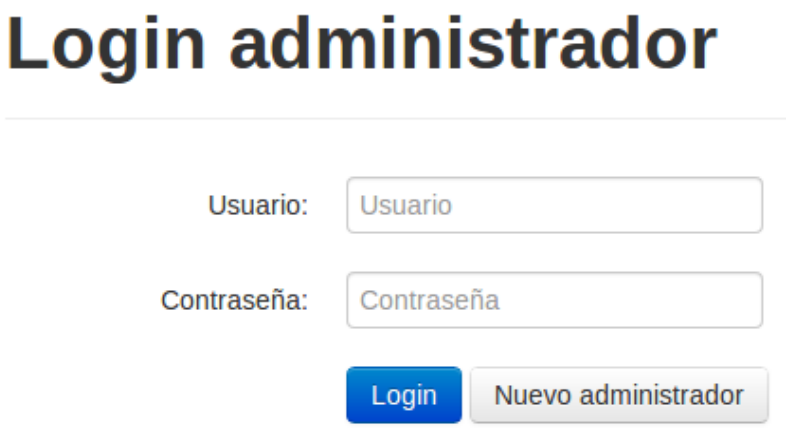


Fig. 60 Pantalla login administrador

2. Nuevo Administrador

Para crear un nuevo administrador se deberá de pulsar el botón **Nuevo Administrador** situado en el Login Administrador, una vez pulsado se accederá a la pantalla **Alta nuevo administrador**.

En la pantalla **Alta nuevo administrador** se deberán introducir los siguientes datos:

- Usuario
- Contraseña
- Nombre y apellido

Una vez introducidos los datos se deberá pulsar el botón **Alta administrador** y, si los datos son correctos, se accederá a la pantalla Login Administrador donde deberá introducirse los nuevos datos para acceder al Panel de Control.

En caso de que ocurra algún error se mostrará una alerta.

Fig. 61 Pantalla alta nuevo administrador

3. Pantalla Principal

La Pantalla Principal del Panel de Control es la pantalla a la que se accede después de introducir el usuario y la contraseña y siempre se retornará a ella pulsando en la palabra *Matroid*, que está en la parte superior izquierda.

En esta pantalla se visualizará un menú lateral, con los diferentes elementos que se pueden controlar, y la sección de búsqueda del elemento Paquete, por ser a priori el elemento más importante.

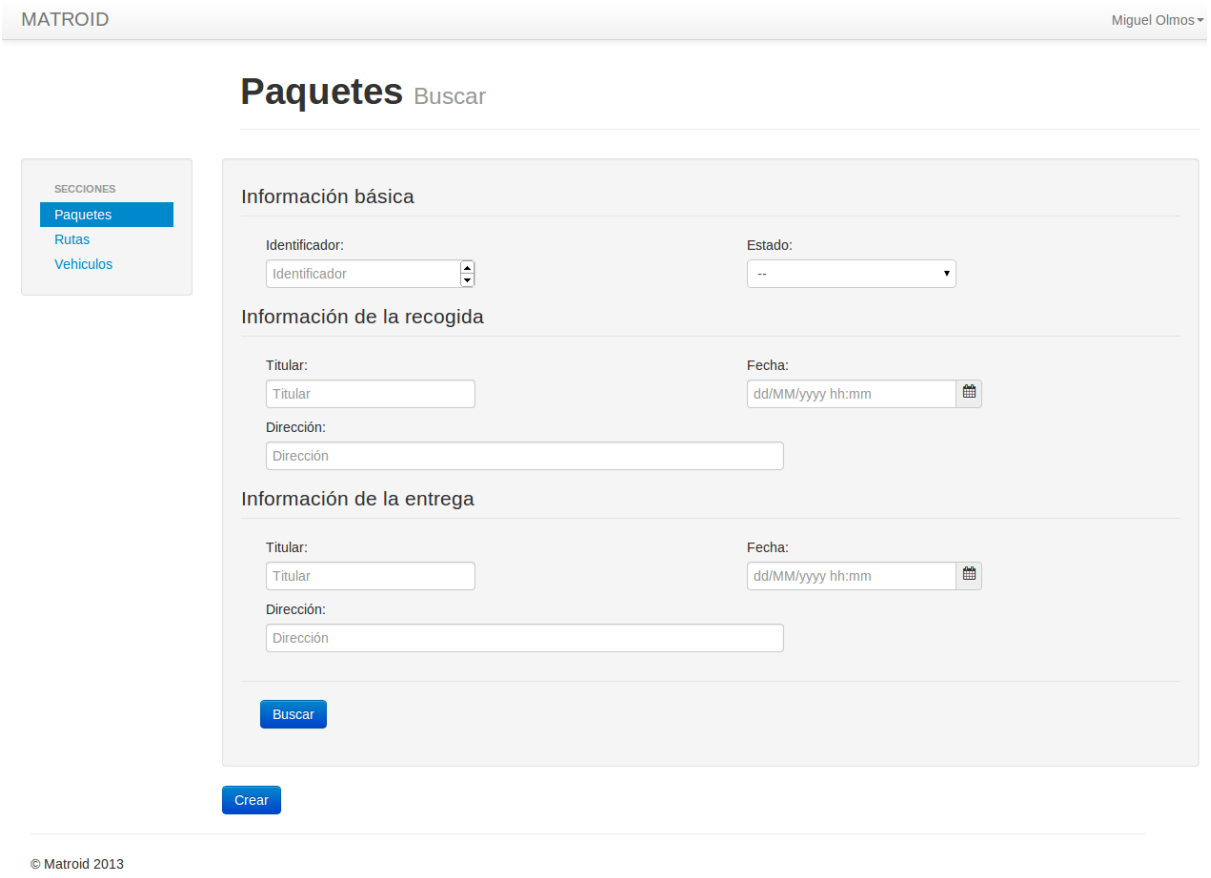


Fig. 62 Pantalla principal panel de control

4. Salir del panel de control

Para salir de forma correcta del panel de control, una vez que se haya accedido como administrador, deberá clicarse sobre el nombre en la parte superior derecha y después clicar en **Salir**.

Fig. 63 Dropdown administrador

5. Editar administrador

Para editar la información de un administrador, se deberá de acceder al Panel de Control introduciendo el usuario y después clicar en el nombre que está en la parte superior derecha. En el menú desplegable que surgirá, se debe de pulsar **Editar**. Una vez pulsado, se accederá a la pantalla de edición de administrador.

En la pantalla de edición de administrador se podrá editar el nombre y apellido y/o la contraseña. Para ello se pueden modificar los campos o introducir la nueva contraseña, en caso de querer cambiarla, y pulsar el botón **Guardar**. Si los cambios se han realizado correctamente se le redirigirá a la pantalla principal del Panel del Control.

En caso de que ocurra algún error se mostrará una alerta.

Fig. 64 Pantalla edición administrador

6. Eliminar administrador

Para eliminar un administrador se deberá de acceder a la pantalla de edición del administrador y pulsar **Eliminar**. Se le mostrará un diálogo de confirmación por si se ha clicado accidentalmente.

Una vez eliminado, se accederá a la pantalla de Login Administrador.

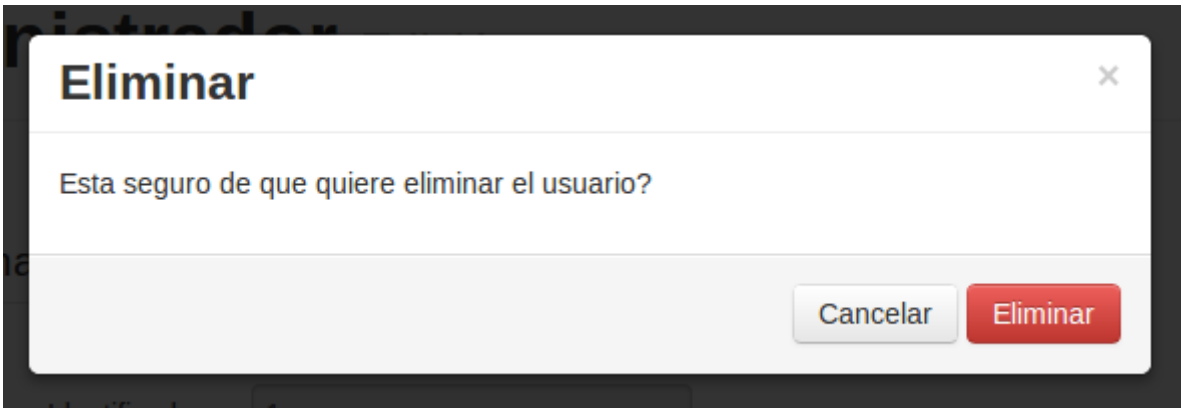


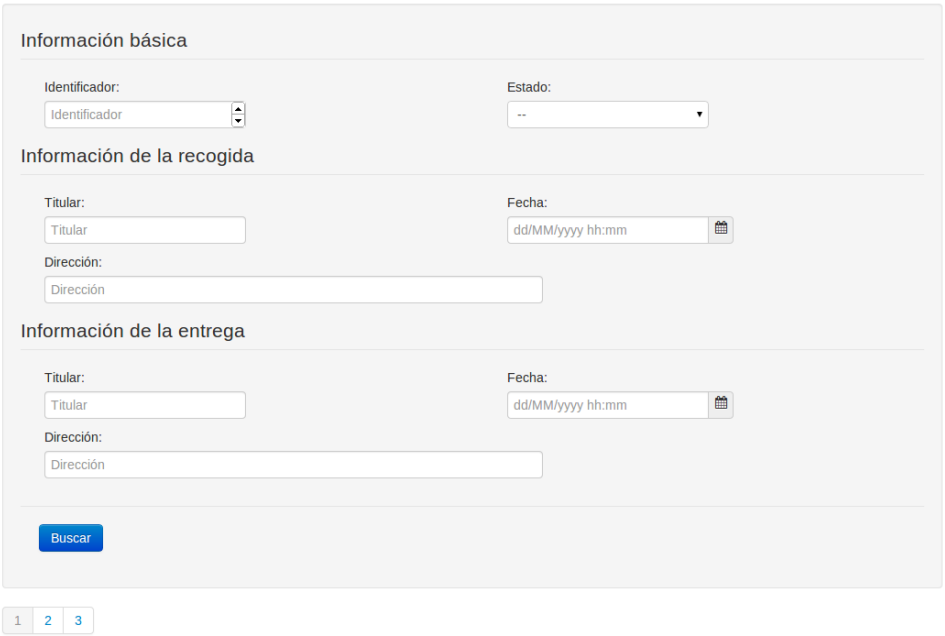
Fig. 65 Modal eliminar administrador

7. Búsqueda de elementos

Para realizar la búsqueda de un elemento, se seleccionará el tipo de elemento que se desea buscar en el menú lateral izquierdo y se rellenarán los criterios sobre los cuales se desea buscar. A continuación, se pulsará el botón **Buscar**.

Los resultados de la búsqueda se mostrarán debajo de los criterios, con un resumen de los datos de cada elemento, y en una paginación de 10 elementos cada página. Para obtener todos los datos de un elemento de la búsqueda, se pulsará sobre el icono de la derecha del resultado, situado en la columna **Detalle**.

Búsqueda del elemento paquete



Identificador	Estado	Titular recogida	Titular entrega	Detalle
1	Entregado	Patricia Sanz Perera	Albert Montserrat Blanco	
2	Entregado	Soledad Martínez Puig	Eloi Gutierrez Maldonado	
3	Entregado	Anna Muñoz Pérez	Juan Luís Ruiz Amado	
4	Entregado	Miguel Ángel Mateo Soler	Ángela García Blázquez	
5	Entregado	Irene López Ferrer	Maribel Estévez Navarrete	
6	Entregado	Margarita Serrano González	Oscar Olmos Padilla	
7	Entregado	Sandra Cobo Moreno	Sonia Valls Barrera	
8	Entregado	Manuel Caballero Pino	Jesús Rodríguez López	
9	Recogido	Alberto Gálvez García	Mª Pilar Labodia Utre	
10	Pendiente de recoger	Juan Antonio Gonzalvo Rito	Javier Sánchez Méndez	

Fig. 66 Pantalla búsqueda paquete

Búsqueda del elemento ruta

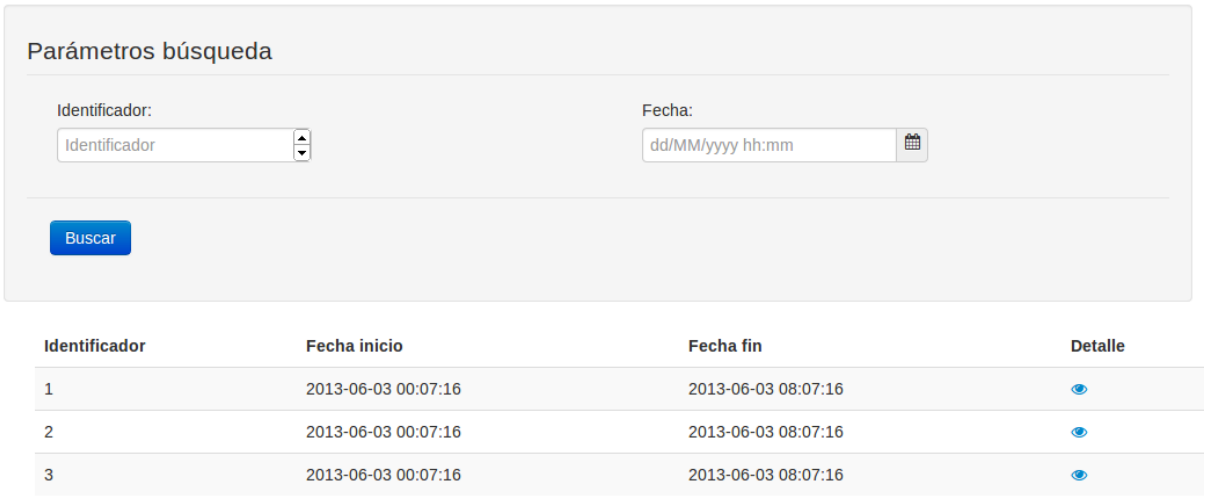


Fig. 67 Pantalla búsqueda ruta

Búsqueda del elemento vehículo

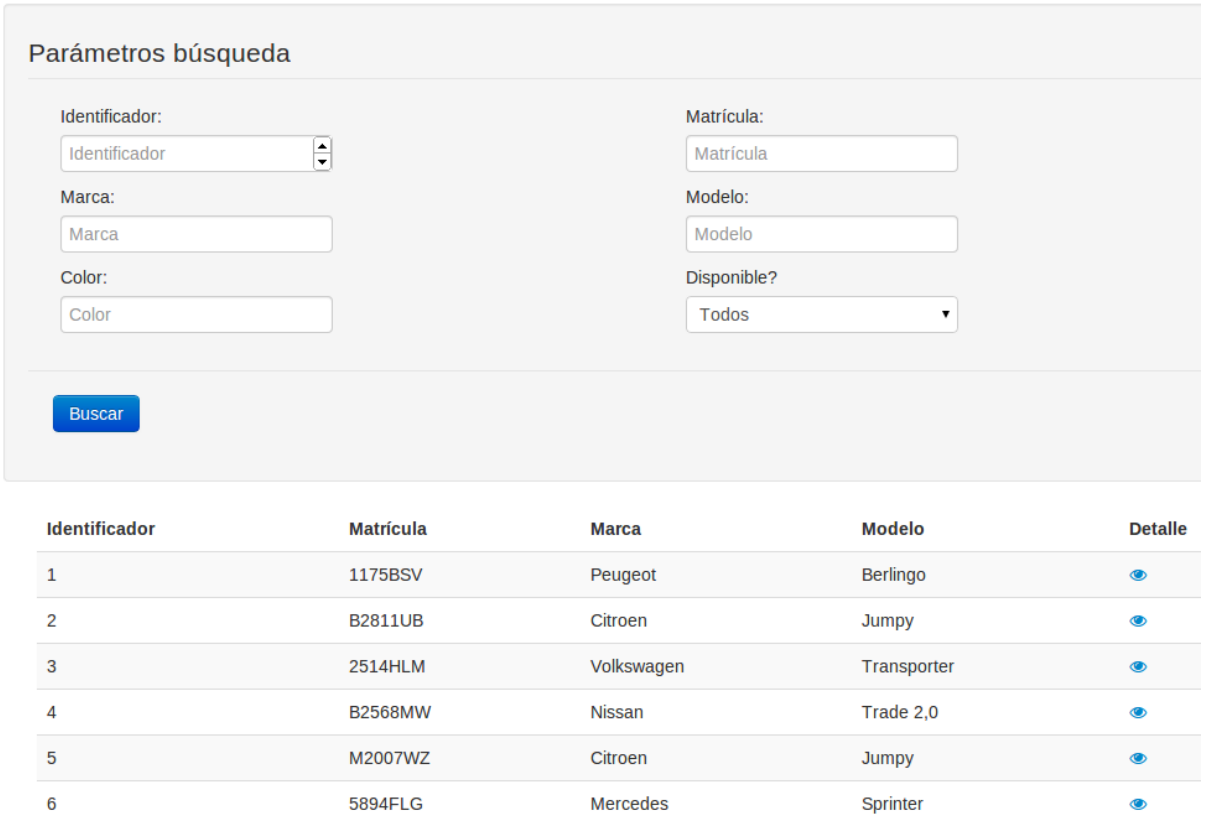


Fig. 68 Pantalla búsqueda vehículo

8. Detalle de elemento

Al acceder al detalle de un elemento a través de la búsqueda, se le mostrarán todos los datos almacenados sobre ese elemento. Además, se le ayudará en la visualización de los datos con mapas siempre que sea posible.

Detalle del elemento paquete

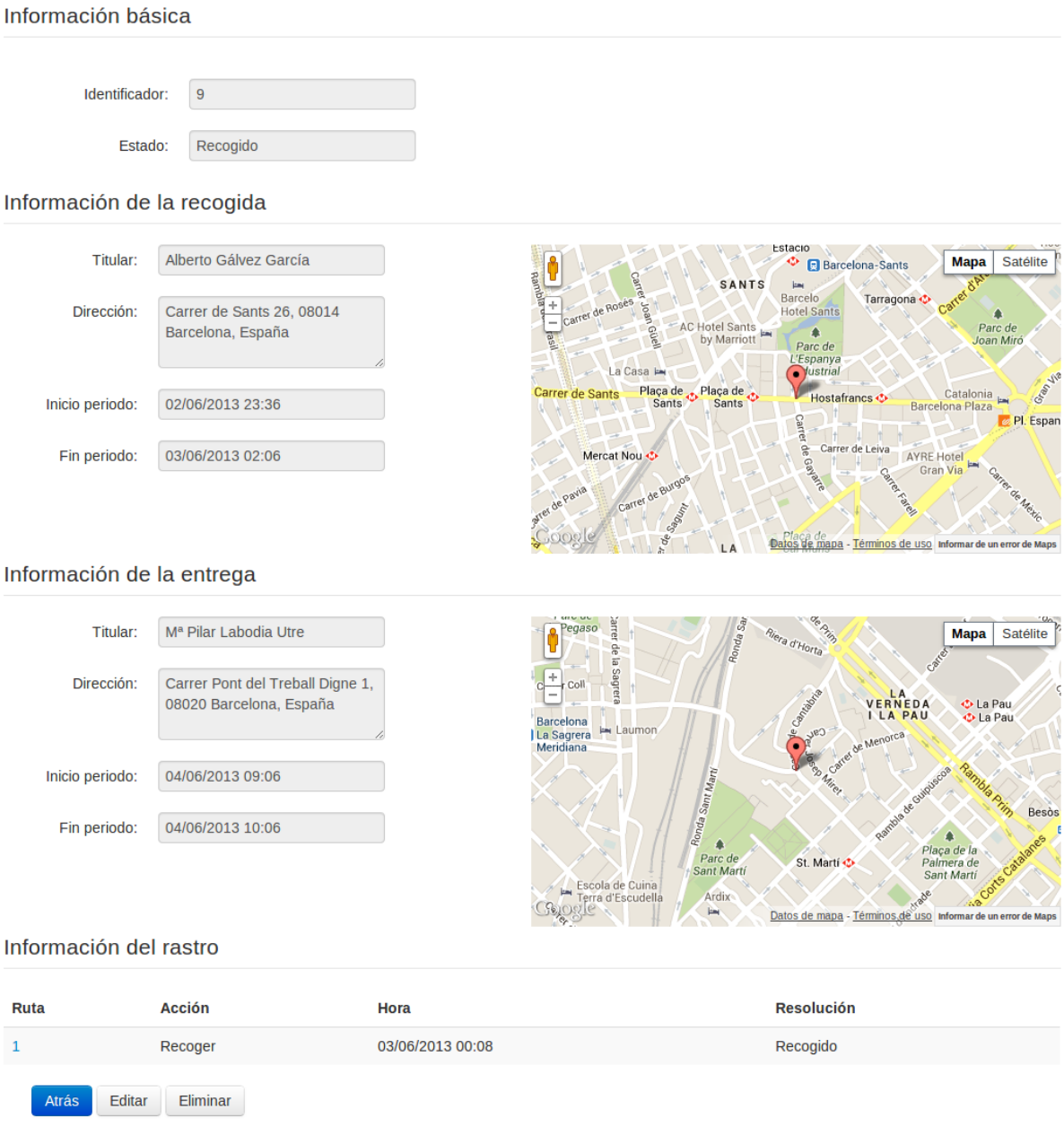


Fig. 69 Pantalla detalle paquete

Detalle del elemento ruta

En el caso concreto de las rutas, al pulsar sobre una parada se desplegará la información de los paquetes de esa parada. Además, como la base desde donde se inicia la ruta es la misma que donde se finaliza, en el mapa se solaparán el primer marcador con el último.

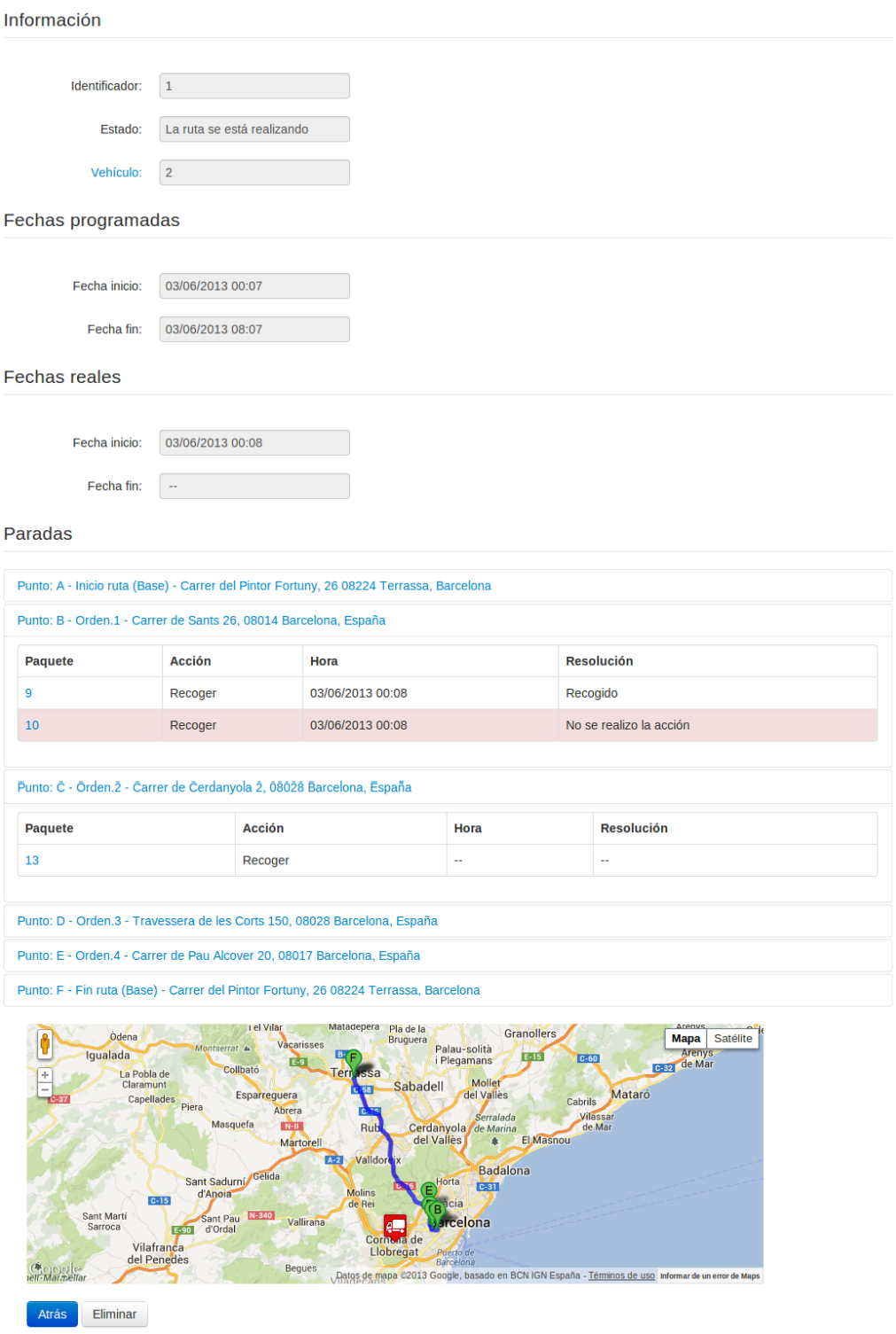


Fig. 70 Pantalla detalle ruta

Detalle del elemento vehículo

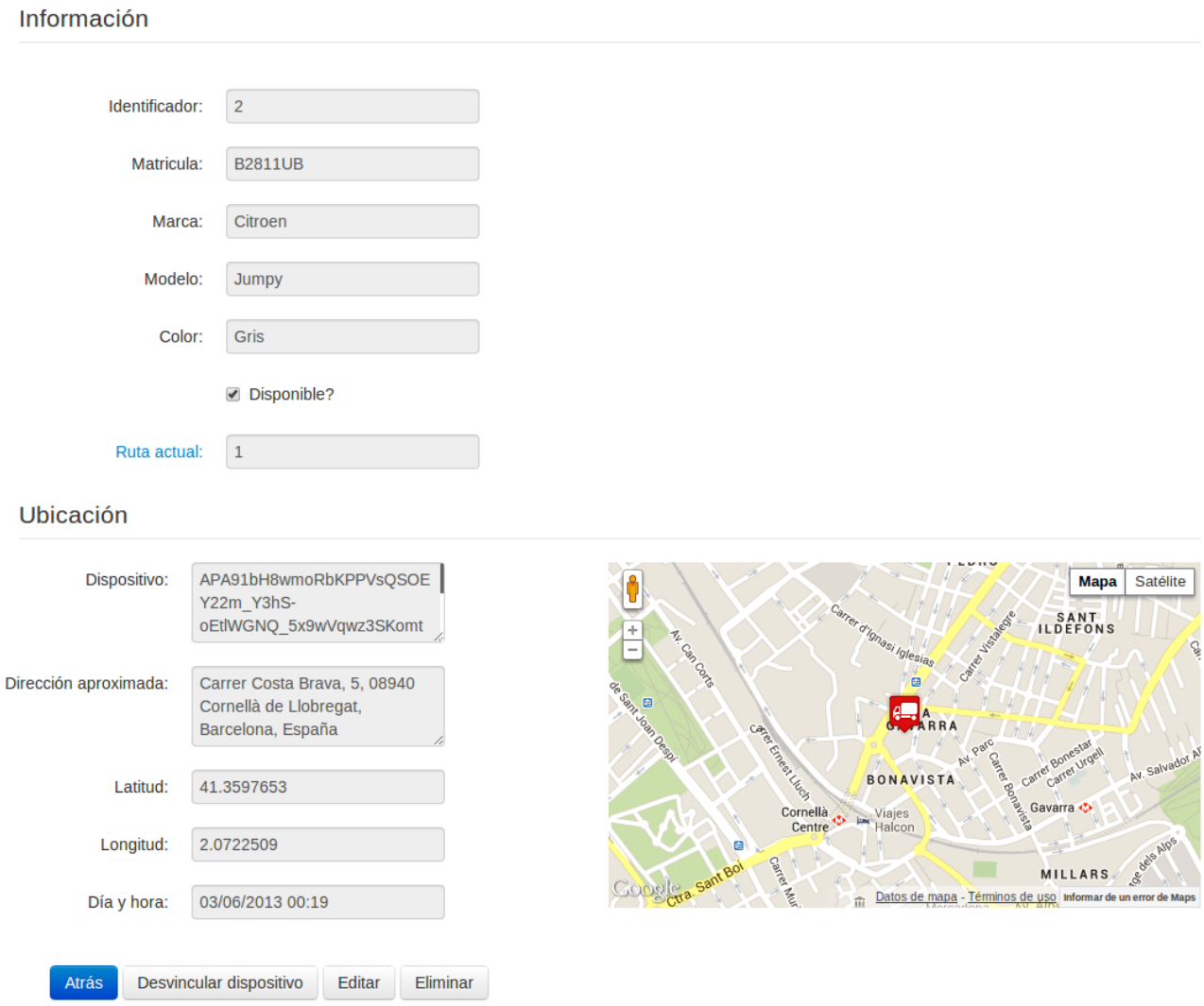


Fig. 71 Pantalla detalle vehículo

9. Edición elemento

Para editar un elemento (sólo se pueden editar paquetes y vehículos) se deberá de acceder al detalle del elemento y pulsar el botón **Editar**, que está situado al final de la página.

Una vez se accede a la pantalla de edición del elemento, se deberán de modificar los datos deseados y pulsar sobre el botón **Guardar**. Si se ha modificado correctamente se accederá al detalle del elemento; en caso contrario, se mostrará una alerta con el error ocurrido.

Edición del elemento paquete

Información básica

Identificador: 9

Estado: Recogido

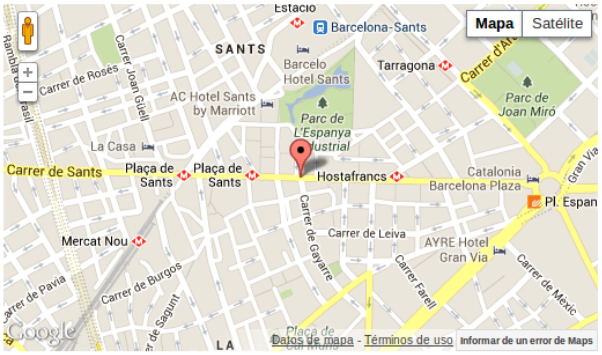
Información de la recogida

Titular: Alberto Gálvez García

Fecha Inicio: 02/06/2013 23:36

Fecha Fin: 03/06/2013 02:06

Dirección: Carrer de Sants 26, 08014 Barcelona, España



Información de la entrega

Titular: Mª Pilar Labodia Utre

Fecha Inicio: 04/06/2013 09:06

Fecha Fin: 04/06/2013 10:06

Dirección: Carrer Pont del Treball Digne 1, 08020 Barcelona, España

Guardar Atrás

Fig. 72 Pantalla edición paquete

Edición del elemento vehículo

Fig. 73 Pantalla edición vehículo

10. Eliminación elemento

Para eliminar un elemento se deberá acceder al detalle del elemento y pulsar el botón **Eliminar**, que está situado al final de la página. Se mostrará un diálogo de confirmación por si se ha clicado accidentalmente.

Si se ha eliminado correctamente, se accederá a la búsqueda de los elementos; en caso contrario, se mostrará una alerta con el error ocurrido.

Fig. 74 Modal eliminación elemento

11. Creación elemento

Para crear un elemento (sólo se pueden crear paquetes y vehículos) se deberá seleccionar el tipo de elemento que se desea crear en el menú de la izquierda y pulsar **Crear**, situado en la parte inferior de la pantalla.

Una vez situados en la pantalla de creación del elemento, se rellenarán los datos necesarios y se pulsará **Guardar**. Si se ha creado correctamente, se accederá al detalle del nuevo elemento, en caso contrario se mostrará una alerta indicando el error.

Creación del elemento paquete

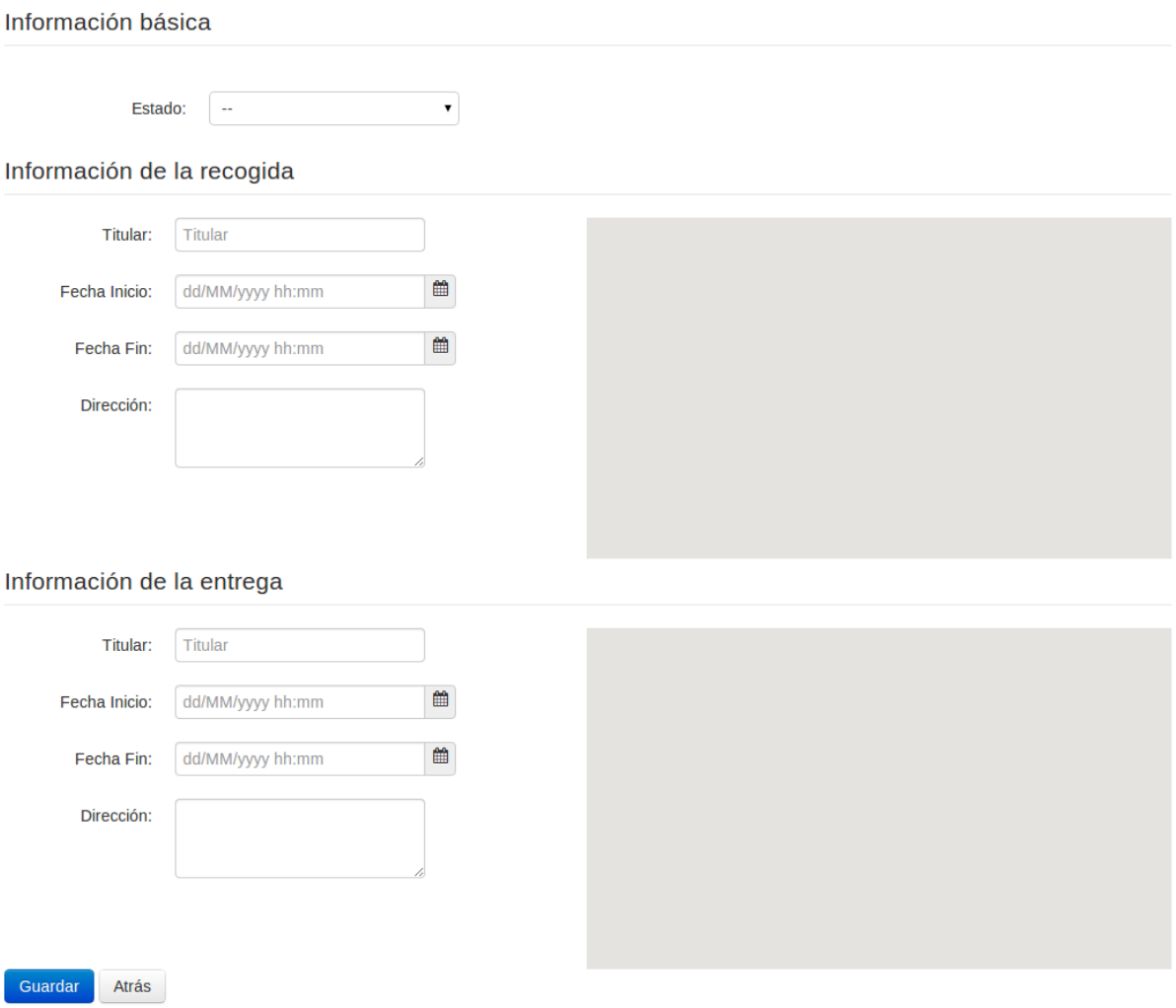


Fig. 75 Pantalla creación paquete

Creación del elemento vehículo

Fig. 76 Pantalla creación vehículo

12. Desvincular vehículo

Los vehículos se pueden desvincular del dispositivo móvil desde el Panel de Control.

Para ello se deberá acceder al detalle del vehículo en cuestión y pulsar **Desvincular dispositivo**. Se mostrará un diálogo de confirmación por si se ha clicado accidentalmente.

Si se desvincula correctamente, se actualizará el campo **Dispositivo** y quedará vacío.

Fig. 77 Modal desvincular vehículo

13. Introducción de direcciones

Cuando se introduzca o modifique una dirección, se tendrá un mapa asociado que se irá actualizando mientras se escribe. Es responsabilidad del administrador asegurarse que la dirección introducida se corresponde con la dirección indicada en el mapa; de no ser así, podría haber errores al realizar las rutas.

14. Botón atrás

En todas las pantallas del Panel de Control se dispone del botón **Atrás**, mediante el cual se podrá cancelar una creación o edición de elemento. En el caso del detalle del elemento, el botón **Atrás** retornará a la búsqueda realizada con los criterios establecidos.

15. Definición de campos

A continuación, se definirán los campos de los diferentes elementos:

Paquete

- Identificador: Identificador asociado al paquete.
- Estado: Estado del paquete.
- Titular recogida: Titular al que se le recogerá el paquete.
- Dirección recogida: Dirección donde se ejecutará la recogida.
- Fechas inicio-fin recogida: Define la ventana de tiempo en la que el titular quiere que se recoja el paquete.
- Titular entrega: Titular al que se le entregará el paquete.
- Dirección entrega: Dirección donde se ejecutará la entrega.
- Fechas inicio-fin entrega: Define la ventana de tiempo en la que el titular quiere que se entregue el paquete.
- Información del rastro: Todas las rutas en las que se ha visto involucrado el paquete.

Ruta

- Identificador: Identificador de la ruta.
- Estado: Estado de la ruta.
- Vehículo: Vehículo que realiza la ruta.
- Fechas inicio-fin programadas: Son las fechas programadas para realizar la ruta.
- Fechas inicio-fin reales: Son las fechas reales en las que se ha realizado la ruta.
- Paradas: Lista con las paradas de la ruta.

Vehículo

- Identificador: Identificador del vehículo.
- Matrícula: Matrícula del vehículo.

- Marca: Marca del vehículo.
- Modelo: Modelo del vehículo.
- Color: Color del vehículo.
- Disponible: Indica si se desea que el vehículo esté disponible para el próximo cálculo de rutas iniciales.
- Ruta actual: Indica la ruta que está ejecutando el vehículo.
- Dispositivo: Identificador del dispositivo móvil vinculado al vehículo.
- Dirección aproximada: Dirección aproximada de su última ubicación conocida.
- Latitud-longitud: Latitud y longitud de su última ubicación conocida.
- Día y hora: Día y hora de su última ubicación conocida.

6.2.3 Usuario del servicio

Se especificará el manual que debe de seguir el usuario de la empresa de paquetería, que utiliza el sistema Matroid, para buscar información sobre el paquete.

Se supone que la página de seguimiento está situada en la misma máquina que utiliza el usuario del servicio de paquetería, de forma que la dirección del servidor será *localhost*.

1. Acceso

Para acceder a la página de seguimiento se deberá introducir en un navegador web la siguiente dirección: http://localhost/MatroidWeb/

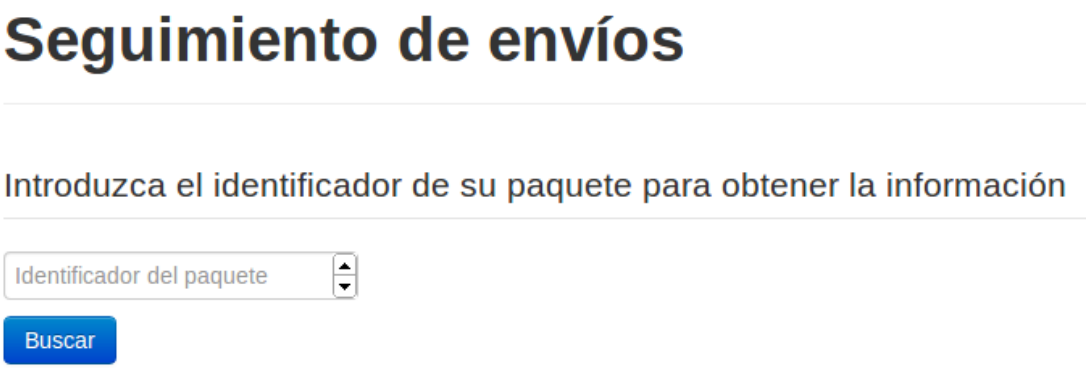


Fig. 78 Pantalla principal página de seguimiento

2. Consulta paquete

Una vez accedido a la página de seguimiento, se deberá introducir el identificador del paquete para obtener toda la información almacenada. La información que se mostrará es la siguiente:

- Identificador: Identificador del paquete.
- Estado: Estado del paquete.
- Titular recogida: Titular de recogida del paquete.
- Dirección recogida: Dirección de recogida del paquete.
- Fechas inicio-fin recogida: Ventana de tiempo en la que el paquete debe de ser recogido.
- Titular entrega: Titular de entrega del paquete.
- Dirección entrega: Dirección de entrega del paquete.
- Fechas inicio-fin entrega: Ventana de tiempo en la que el paquete debe de ser entregado.
- Información del rastro: Todas las rutas en las que se ha visto involucrado el paquete.

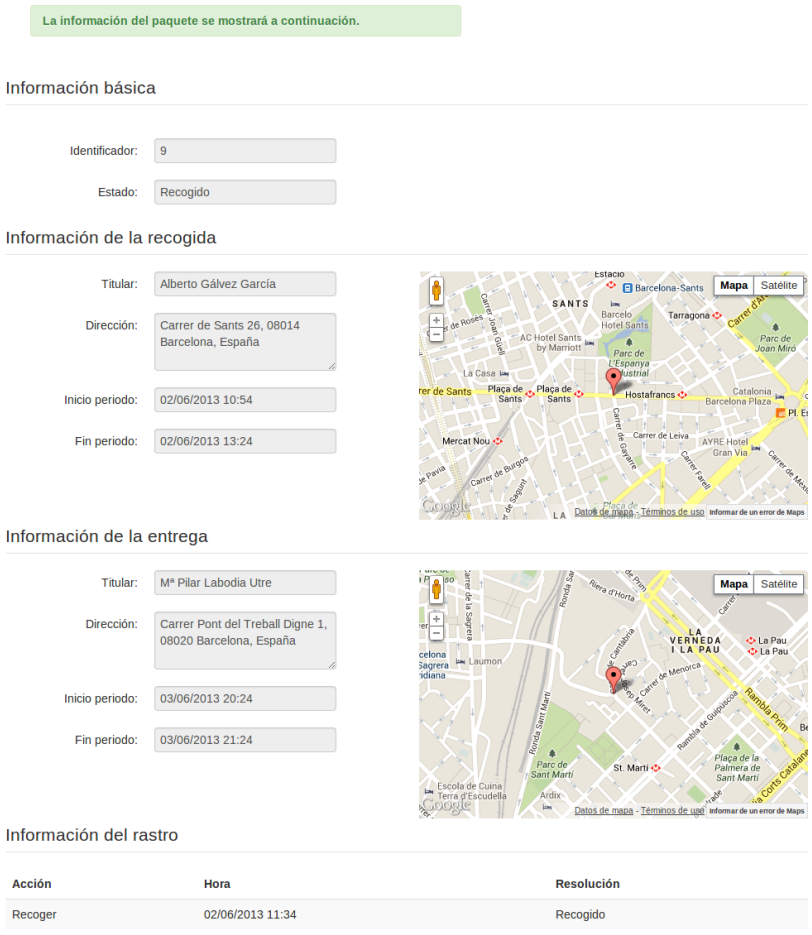


Fig. 79 Pantalla resultado búsqueda página de seguimiento

7 Gestión del proyecto

7.1 Planificación

A continuación, se muestra el diagrama de Gantt inicial del proyecto:

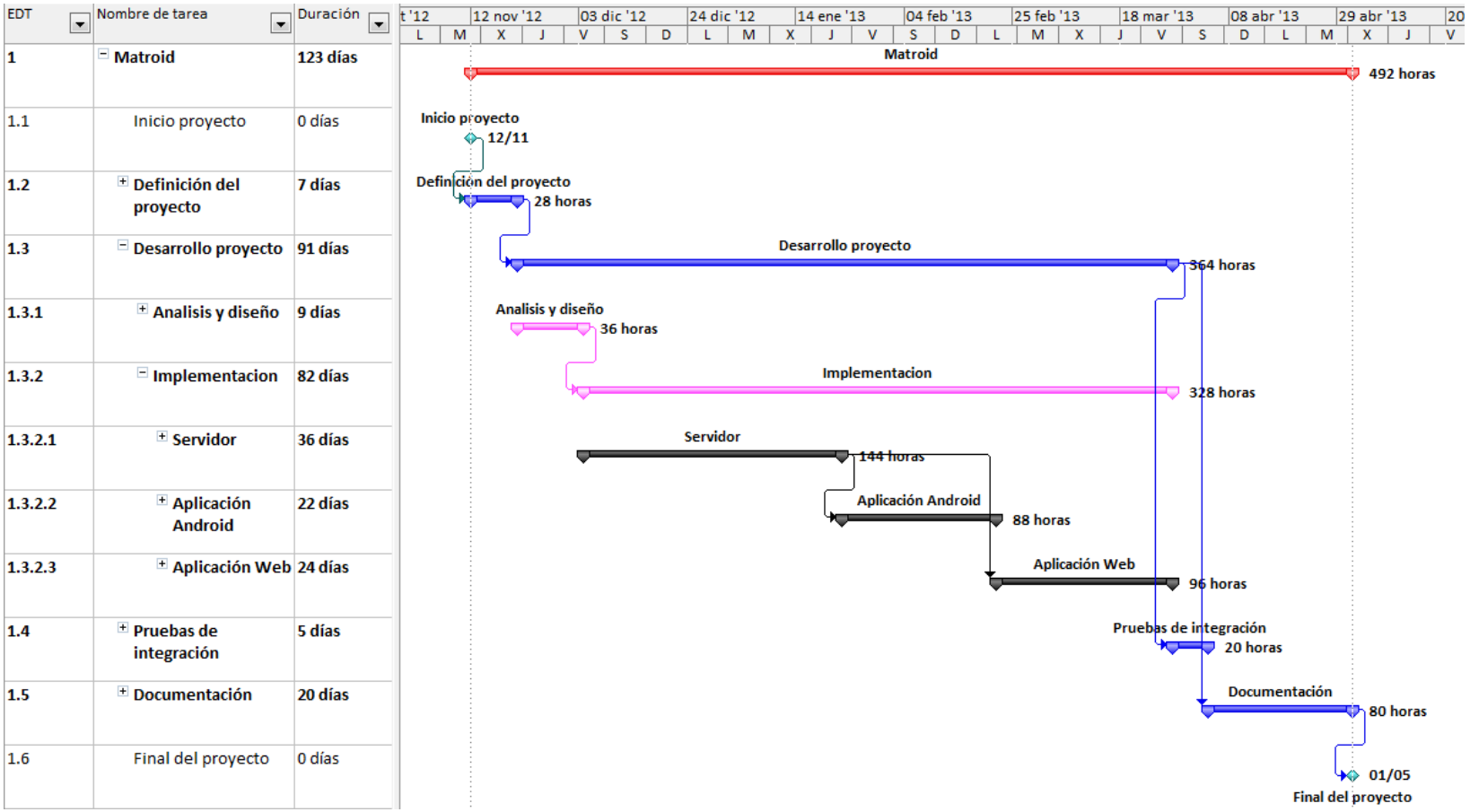


Fig. 80 Diagrama Gantt inicial

7.2 Fases del proyecto

En este apartado se definen las fases del proyecto que se ha realizado y que están reflejadas en el Gantt.

7.2.1 Definición del proyecto

En esta fase he definido a grandes rasgos como debería ser el sistema Matroid y los objetivos que debería cumplir.

Se sabía que es un sistema que funcionaría utilizando dispositivos móviles Android, mediante los cuales se conectaría a un servidor para obtener la información de la ruta calculada. Estos dispositivos mostrarían diferentes informaciones y el recorrido de la siguiente parada. Además, permitirían indicar cuando un paquete se ha recogido o entregado.

Otra característica que se definió la posibilidad de actualización de la ruta, de forma que se recalcule la ruta cada cierto tiempo y, en caso de que se encuentre una más óptima, se modifique y avise al dispositivo móvil.

También se estableció la utilización de un servidor externo de ámbito matemático, que no ha entrado en el alcance de este proyecto, que ejecutaría las operaciones más complejas para el cálculo de la ruta óptima.

Otro objetivo que se ha definido, es la creación de un panel de control y una página de seguimiento que permitirían visualizar, crear, editar y eliminar los elementos del sistema.

Por último se definieron los lenguajes mediante los cuales se realizaría el sistema Matroid y el método de comunicación con el servidor externo.

7.2.2 Desarrollo del proyecto

Esta fase representa la fase más duradera de todo el proyecto y es en la que se realizó el proyecto propiamente dicho. Se puede dividir en dos grandes bloques:

Análisis y diseño

En este bloque se definió el camino que se iba a recorrer y las herramientas que se iban a utilizar para lograr alcanzar los objetivos. Además, se buscaron soluciones a los posibles problemas que en un principio se nos iban a presentar.

Una de las tareas de este bloque fue definir el método de comunicación entre los dispositivos móviles y el servidor central; así como la arquitectura, que seguiría el código Java, para mantener un patrón que separara el acceso a base de datos, el dominio y los servicios a realizar por el programa.

Para el acceso a base de datos se pensó primeramente en la utilización del framework Hibernate, pero se desestimó la propuesta por ser un framework algo

complejo para los objetivos que se querían alcanzar. Finalmente se optó por una capa de DAOs mediante la cual modularizaríamos el acceso a base de datos.

Un problema, al que se le encontró solución, fue la creación y actualización de las rutas; ya que al realizarse el cálculo en un servidor externo de ámbito matemático, no podía hacer referencia a rutas, vehículos y/o paquetes para obtener, por ejemplo, una actualización de la ruta.

También se buscó un framework para la implementación del panel de control y la página de seguimiento. Primeramente se encontró el framework CodeIgniter y más adelante, se decidió añadir el framework Bootstrap para implementar la capa visual.

Implementación

En este bloque se ha implementado el sistema Matroid desde el punto de vista técnico ya que se escribió todo el código necesario para alcanzar los objetivos propuestos.

En la implementación se comenzó por la arquitectura del código Java, que separa el dominio, de los servicios y del acceso a base de datos, de forma que nos creara una base sólida sobre la cual programar de forma más fácil. El siguiente punto fue implementar el servidor central en el cual se encuentra la gran lógica del sistema, como el cálculo inicial y la actualización de las rutas, para más adelante implementar la aplicación Matroid que se apoyaría en el servidor. Por último, se implementó el panel de control, la página de seguimiento y un servidor web service falso que simularía nuestra conexión con el servidor externo de ámbito matemático.

Durante la implementación se tuvo que superar problemas, que no estaban previstos, como el envío de caracteres con acentos a los dispositivos, la incerteza de la frecuencia con la que el sistema de ubicación Android nos informa, el mantenimiento correcto de la pila de actividades de Android o el gran tiempo que se tarda en ejecutar una actualización de ruta… Todos estos problemas fueron resueltos, aunque ampliaron el tiempo calculado para la implementación.

En esta fase también se realizaron las pruebas individuales de cada elemento del sistema Matroid para que en el momento de integrarlos provocaran los mínimos fallos posibles.

Además he adquirido nuevos conocimientos como pueden ser la tecnología push, el framework Bootstrap, los web service o las APIs externas; y he ampliado otros como la programación en Android, Java y PHP, el framework CodeIgniter o la utilización de los Servlets.

Un objetivo que siempre ha estado presente durante la implementación ha sido el de escribir un código claro, comentado y bien modularizado en capas, para que la lectura y el retorno del desarrollo del código en un futuro sea lo más fácil posible.

7.2.3 Pruebas

En esta fase del proyecto se realizaron pruebas lo más completas posibles, ya que intervenían todos los elementos del sistema a la vez. En estas pruebas se simulaban las situaciones que debe de afrontar el sistema en su uso habitual, como es la interacción con varios dispositivos o la actualización de rutas.

Además, una parte de las pruebas fueron destinadas a la simulación de situaciones anómalas que conllevan la utilización de aplicaciones móviles, como puede ser la perdida de conexión del dispositivo móvil. En ningún caso la aplicación podía lanzar un fallo que la cerrara y siempre debía informar al usuario de la situación si era necesario.

Por otro lado, se controlaron y realizaron pruebas sobre los posibles fallos que pueden causar las APIs o el web service externo; ya que, por su naturaleza, no tenemos control sobre la disponibilidad de estos elementos ni de los resultados que nos pueden retornar.

La aplicación Android se probó en diferentes dispositivos; ya que, aunque se programa en Java y la máquina virtual disponible en ellos nos debería de asegurar una ejecución idéntica entre todos, en realidad siempre existen errores que se producen en algunos dispositivos y en otros no.

7.2.4 Documentación

Por último se ha realizado la memoria del proyecto. Aunque la escritura de este se redactó en último lugar, desde un principio fui anotando los pasos que realizaba y todo aquello que creía que debería de estar reflejado en la memoria. Aunque todas esas anotaciones me han servido de ayuda, siempre hay detalles que no anoté y he tenido que revisar el trabajo realizado para presentar una documentación lo más completa posible.

7.3 Presupuesto

A continuación, se definirá el presupuesto necesario para la creación y utilización del sistema Matroid.

7.3.1 Hardware

El hardware mínimo que se debería utilizar para el sistema Matroid se basa principalmente en dos elementos. Un servidor que ejecutará todos los procesos de las rutas, recibirá las peticiones y alojará las páginas web de seguimiento y panel de control; y un dispositivo móvil por cada vehículo que tenga la empresa.

Hardware	Precio
Servidor - HP ProLiant ML310e G8 • Procesador: Intel® Xeon® E3-1220V2 (4 núcleos, 3.1 GHz) • Memoria RAM: 2GB (ampliable hasta 32GB)	540 €
Dispositivo móvil - Samsung Galaxy Mini 2	105 € x unidad

Fig. 81 Tabla presupuesto hardware

7.3.2 Software

La mayoría del software que se ha utilizado para diseñar el sistema Matroid es gratuito; aunque algunos, al ser destinados a uso comercial, requieren un coste económico. Las licencias mínimas que se deben de obtener son:

Software	Precio por año
Google Maps API for Business	$10,000
MySQL Enterprise Edition	$5,000
Total	**$15,000**

Fig. 82 Tabla presupuesto software

7.3.3 Desarrollo del sistema

En este apartado definiremos el coste del desarrollo del sistema Matroid.

El desarrollo se ha divido en tres roles:

- Jefe de proyecto: Encargado de la planificación y especificación del proyecto.
- Analista: Encargado de diseñar la estructura del software y de probarlo.
- Programador: Encargado de programar el software.

Perfil	Precio hora	Horas	Precio total
Jefe de proyecto	80 €/h	28 h	2.240 €
Analista	70 €/h	56 h	3.920 €
Programador	50 €/h	328 h	16.400 €
Total	**-**	**-**	**22.560 €**

Fig. 83 Tabla presupuesto inicial desarrollo del sistema

7.4 Ejecución del proyecto

A continuación, se muestra el diagrama de Gantt del proyecto una vez finalizado:

EDT	Nombre de tarea	Duración
1	Matroid	151,5 días
1.1	Inicio proyecto	0 días
1.2	Definición del proyecto	7 días
1.3	Desarrollo proyecto	116,5 días
1.3.1	Analisis y diseño	9 días
1.3.2	Implementacion	107,5 días
1.3.2.1	Servidor	43 días
1.3.2.2	Aplicación Android	31,75 días
1.3.2.3	Aplicación Web	32,75 días
1.4	Pruebas de integración	8 días
1.5	Documentación	20 días
1.6	Final del proyecto	0 días

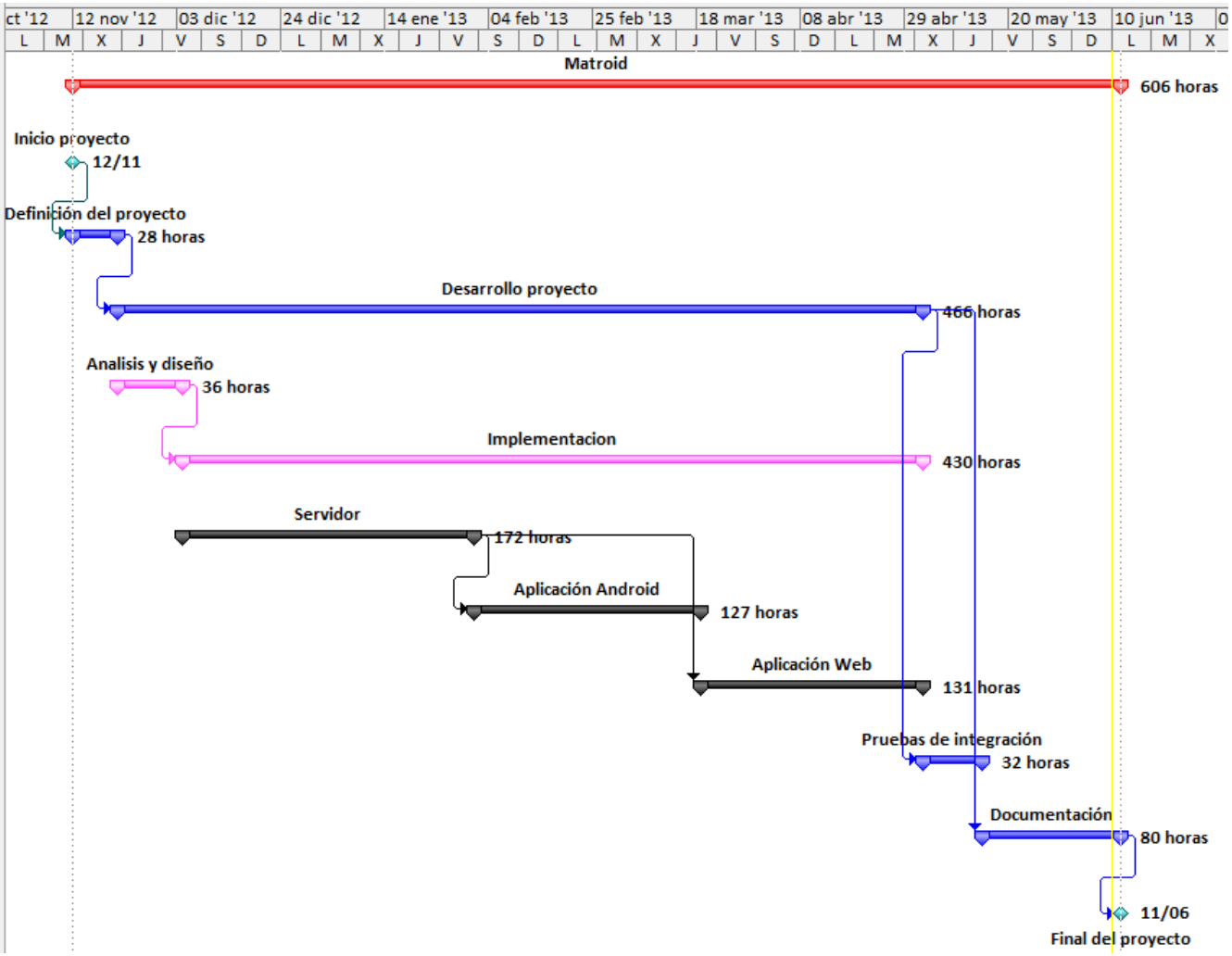


Fig. 84 Diagrama Gantt final

Como podemos observar, la implementación del proyecto se ha visto aumentada unas 114 horas. Este aumento ha sido producido por los siguientes motivos:

- Incremento en el tiempo de las tareas que se habían planificado.
- Cambio de framework en el desarrollo de la aplicación web cuando ya se había desarrollado alguna parte de esta.

Con este incremento de horas, el coste final del desarrollo del sistema Matroid será:

Perfil	Precio hora	Horas	Precio total
Jefe de proyecto	80 €/h	28 h	2.240 €
Analista	70 €/h	68 h	4.760 €
Programador	50 €/h	430 h	21.500 €
Total	**-**	**-**	**28.500 €**

Fig. 85 Tabla presupuesto final desarrollo del sistema

La diferencia entre el presupuesto inicial y final ha sido de:

Presupuestos	Precio
Presupuesto inicial	22.560 €
Presupuesto final	28.500 €
Diferencia	**- 5.940 €**

Fig. 86 Tabla diferencia presupuestos

8 Conclusiones

Una vez que se finaliza todo proyecto, se han de extraer conclusiones de la experiencia adquirida al realizarlo. Son estas conclusiones las que nos evitan cometer los mismos errores en un futuro y mejorar nuestros conocimientos y actuaciones.

Las conclusiones que he podido obtener del proyecto, después de haber trabajado en él durante los últimos meses, son las siguientes:

En primer lugar, el principal objetivo del proyecto, que era generar un pequeño sistema que gestione la actividad principal de una empresa de transporte de paquetería, se ha alcanzado creando un sistema que nos permite optimizar el recorrido de las rutas y, siempre que sea necesario, actualizarlas cuando se están realizando. Además permite registrar la actividad de los paquetes y, mediante un navegador web, consultar, crear, editar y eliminar los datos del sistema.

En segundo lugar, he comprendido la complejidad que puede suponer algunos problemas que, a priori, podrían parecer fáciles. Es el caso del cálculo de rutas que, aún no siendo abordado por el proyecto en su totalidad, he descubierto que puede comprometer muchísimo la viabilidad de un proyecto.

En tercer lugar, me he dado cuenta del gran potencial que tiene el sistema operativo Android, conocido por todos en los dispositivos de ámbito doméstico, en un ámbito laboral, tiene mucho terreno a recorrer que aún no se ha explotado y muchas nuevas posibilidades que ofrecer para las empresas.

Por otro lado, he sufrido uno de los problemas más graves del sistema operativo Android y que es también una de sus grandes virtudes, la versatilidad. En un principio es un valor añadido muy importante a esta plataforma y que, gracias a su máquina virtual, nos garantiza una compatibilidad entre diferentes dispositivos. Pero en realidad no es así, ya que en algunos casos ocurren errores y en otros no. A pesar de esto, este sistema operativo ofrece muchas más ventajas que inconvenientes y es una plataforma muy interesante a considerar por las empresas.

Para finalizar las conclusiones sobre el proyecto, he descubierto que siempre que un sistema debe de interactuar con un mundo que no garantiza la transmisión de datos estables, como el de las conexiones inalámbricas en dispositivos que se encuentran en movimiento, el sistema debe de ser muy cuidadoso y garantizar, si cabe aún más, que los datos no serán corrompidos por una mala o débil comunicación.

A nivel de conocimientos, este proyecto me ha servido para adquirir algunos nuevos que antes ignoraba, como por ejemplo la utilización de la tecnología push o el framework Bootstrap; además de ampliar otros ya alcanzados anteriormente, como pueden ser las programaciones Android, Java y PHP y el framework CodeIgniter.

A un nivel más personal, la experiencia de este proyecto ha sido muy positiva y, aunque durante los meses que le he dedicado ha habido momentos muy estresantes en los que me preguntaba si podría llegar a tiempo, haberlo realizado espero que me dé la experiencia necesaria para afrontar mejor futuros proyectos.

9 Glosario

***Activity*:** Componente de las aplicaciones Android que representa cada una de las interfaces de la pantalla.

Actor: Toda entidad externa al sistema que guarda una relación con este y que le demanda una funcionalidad.

AJAX: Acrónimo de *Asynchronous JavaScript And XML.*

API: Acrónimo de *Application Programming Interface*.

APK: Acrónimo de *Application PacKage File*.

Camino Hamiltoniano: Una sucesión de aristas adyacentes que visita todos los vértices de un grafo una sola vez.

Cron: Administrador regular de procesos en segundo plano que los ejecuta a intervalos regulares.

Dalvik: Máquina virtual que utiliza la plataforma para dispositivos móviles Android.

DAO: Acrónimo de *Data Access Object*.

DOM: Acrónimo de *Document Object Model*.

Framework: Una estructura conceptual y tecnológica de soporte definido que puede servir de base para la organización y desarrollo de software.

GPU: Acrónimo de *GraphicsPprocessing Unit*.

HTTP: Acrónimo de Hypertext Transfer Protocol.

HTTPS: Acrónimo de Hypertext Transfer Protocol Secure.

JSON: Acrónimo de *JavaScript Object Notation*.

JSP: Acrónimo de *Java Server Pages*.

JVM: Acrónimo de *Java Virtual Machine*.

MD5: Acrónimo de *Message-Digest Algorithm 5*.

MVC: Acrónimo de Modelo Vista Controlador.

NP-Completo: Subconjunto de problemas de decisión en NP, de modo que todo problema en NP se puede reducir en cada uno de los problemas de NP-completo.

OpenGL: Acrónimo de *Open Graphics Library*.

Rollback: Operación que devuelve la base de datos a algún estado previo.

Ruta inicial: Primer cálculo del recorrido de una ruta.

SALT: Cadena de caracteres que se le añade a las contraseñas antes de ser cifradas para aumentar la seguridad.

Servlet: Objetos que corren dentro y fuera del contexto de un contenedor de Servlets.

SHA1: Acrónimo de *Secure Hash Algorithm*.

Toast: Pequeño pop-up que se utiliza para mostrar información durante un corto periodo de tiempo en las aplicaciones Android.

Web service: Tecnología que utiliza un conjunto de protocolos y estándares para intercambiar datos entre aplicaciones

XML: Acrónimo de *eXtensible Markup Language*.

XSS: Acrónimo de *Cross-Site Scripting*.

10 Bibliografía

Documentación

[1] Hussein Elbaroudy, Gonzalo Soler: “Curso de introducción a Android”. 2011.

[2] JEDI: “PHP Avanzado”. 2011.

[3] Departament d’enginyeria de serveis i sistemes d’información: “Base de dades (BD)”. 2010.

[4] Antoni Lozano: “Algorísmia, calcilabilitat i complexitat (ALCC)”. 2010.

Páginas web

[5] Wikipedia http://www.wikipedia.org/

[6] W3schools http://www.w3schools.com/

[7] Android Developer http://developer.android.com

[8] PHP http://www.php.net/

[9] MySQL http://www.mysql.com/

[10] CodeIgniter http://ellislab.com/codeigniter/

[11] Bootstrap http://twitter.github.io/bootstrap/

[12] Bootstrap-Datetimepicker http://tarruda.github.io/bootstrap-datetimepicker/

[13] API Google Maps https://developers.google.com/maps/?hl=es

[14] Gson https://code.google.com/p/google-gson/

[15] jQuery http://jquery.com/

[16] Tomcat http://tomcat.apache.org/

[17] log4j http://logging.apache.org/log4j/1.2/

[18] Load Impact http://loadimpact.com/

[19] Stackoverflow http://stackoverflow.com/

[20] GCM http://www.javahispano.org/android/2012/8/8/mecanismos-para-el-envio-de-notificaciones-a-los-usuarios-de.html

[21] Firma aplicaciones http://androideity.com/2011/08/25/%C2%BFcomo-firmar-aplicaciones-android/

[22] Mapa Android http://www.sgoliver.net/blog/?p=3244

[23] Marcadores Google Maps http://www.funcion13.com/2012/05/07/google-maps-marcadores/

[24] Expresiones regulares http://www.intergraphicdesigns.com/blog/2012/06/05/rapido-y-completo-expresiones-regulares-en-php/

[25] Expresiones regulares http://www.codigogratis.com.ar/expresiones-regulares-en-php/

[26] Web Service http://renewedordead.blogspot.com.es/2012/04/crear-un-web-service-con-eclipse.html

[27] Cliclo vida Activity http://telekita.wordpress.com/2012/02/03/ciclo-de-vida-de-una-activity/

[28] Fechas Java http://carloszuluaga.wikidot.com/articulos:manejo-de-fechas-en-java-ii

[29] Threads http://www.reloco.com.ar/prog/java/threads.html

[30] ListView http://www.nosinmiubuntu.com/2012/11/items-personalizados-para-un-listview.html

[31] XML http://aprendiendo-software.blogspot.com.es/2012/01/como-escribir-archivo-xml-en-java-jdom.html

[32] Flags Intent Android http://blog.akquinet.de/2010/04/15/android-activites-and-tasks-series-intent-flags/

11 Anexo: XMLs

A continuación, se definirá el formato de los XMLs que se envían y reciben del web service para realizar el cálculo inicial o actualización de la ruta.

11.1 Cálculo inicial

11.1.1 Formato de envío

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Este es el XML de envío al web service para el cálculo inicial de las rutas.
-->
<xml>
	<!-- Se especifica el tipo de petición -->
	<peticion>petición(calculoInicial)</peticion>
	<!-- Se especifica las fechas inicio y fin de las rutas a crear -->
	<dateInicio>dateInicio</dateInicio>
	<dateFin>dateFin</dateFin>
	<!-- Se indica los vehículos disponibles para el cálculo -->
	<vehiculo>
		<id>id</id>
	</vehiculo>
	<vehiculo>
		...
	</vehiculo>
	<!-- Se especifica las distancias y los tiempos estimados entre la base y
	todos los paquetes disponibles para el cálculo -->
	<origen>
		<distancias>
			<destino>
				<distancia>distancia(metros)</distancia>
				<duracion>tiempoEstimado(segundos)</duracion>
				<id>idPaqueteDestino</id>
			</destino>
			<destino>
				...
			</destino>
		</distancias>
	</origen>
	<!-- Por cada paquete se especifica la fecha incio y fin de la recogida o
	entrega; y las distancias y los tiempos estimados entre el paquete y el
	restro de paquetes disponibles para el cálculo -->
	<paquete>
		<id>id</id>
		<fechaInicio>fechaInicio</fechaInicio>
		<fechaFin>fechaFin</fechaFin>
		<distancias>
			<destino>
				<distancia>distancia(metros)</distancia>
				<duracion>tiempoEstimado(segundos)</duracion>
				<id>idPaqueteDestino</id>
			</destino>
			<destino>
				...
			</destino>
			<destino>
```

```
                        <!-- El id=0 indica la distancia y el tiempo estimado
                        entre el paquete y la base -->
                        <id>0</id>
                        ...
                  </destino>
            </distancias>
      </paquete>
      <paquete>
            ...
      </paquete>
</xml>
```

11.1.2 Formato de respuesta

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Este es el XML que recibiremos del webservice del cálculo inicial de las
rutas.-->
<xml>
      <!-- Se especifica el tipo de petición -->
      <peticion>petición(calculoInicial)</peticion>
      <!-- Nos indicará cada ruta a realizar, el vehículo que lo debe de
      realizar, las paradas que contiene ordenadas y que paquetes se tiene que
      entregar o recoger en cada parada. -->
      <ruta>
            <vehiculo>
                  <id>id</id>
            </vehiculo>
            <parada>
                  <idPaquete>id</idPaquete>
                  <idPaquete>...</idPaquete>
            </parada>
            <parada>
                  ...
            </parada>
      </ruta>
      <ruta>
            ...
      </ruta>
</xml>
```

11.2 Actualización

11.2.1 Formato de envío

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Este es el XML de envío al web service para la actualización de una ruta. -->
<xml>
	<!-- Se especifica el tipo de petición -->
	<peticion>petición(actualizacion)</peticion>
	<!-- Se especifica la fecha actual -->
	<dateActual>dateActual</dateActual>
	<!-- Se especifica las fechas inicio y fin de la ruta -->
	<dateInicio>dateInicio</dateInicio>
	<dateFin>dateFin</dateFin>
	<!-- Se especifica la fecha inicio real de la ruta -->
	<dateInicioReal>dateInicioReal</dateInicioReal>
	<!-- Se especifica las distancias y los tiempos estimados entre la posición
	actual del vehículo y todos los paquetes que faltan por realizar -->
	<origen>
		<distancias>
			<destino>
				<distancia>distancia(metros)</distancia>
				<duracion>tiempoEstimado</duracion>
				<id>idPaqueteDestino</id>
			</destino>
			<destino>
				...
			</destino>
			<destino>
				<!-- El id=0 indica la distancia y el tiempo estimado
				entre la posición actual del vehículo y la base -->
				<id>0</id>
				...
			</destino>
		</distancias>
	</origen>
	<!-- Por cada paquete se especifica la fecha incio y fin de la recogida o
	entrega; y las distancias y los tiempos estimados entre el paquete y el
	restro de paquetes que faltan por realizar -->
	<paquete>
		<id>id</id>
		<fechaInicio>fechaInicio</fechaInicio>
		<fechaFin>fechaFin</fechaFin>
		<distancias>
			<destino>
				<distancia>distancia(metros)</distancia>
				<duracion>tiempoEstimado(segundos)</duracion>
				<id>idPaqueteDestino</id>
			</destino>
			<destino>
				...
			</destino>
			<destino>
				<!-- El id=0 indica la distancia y el tiempo estimado
				entre el paquete y la base -->
				<id>0</id>
				...
			</destino>
```

```
            </distancias>
      </paquete>
      <paquete>
            ...
      </paquete>
</xml>
```

11.2.2 Formato de respuesta

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Este es el XML que recibiremos del webservice de la actualización de una
ruta. -->
<xml>
      <!-- Se especifica el tipo de petición -->
      <peticion>petición(actualizacion)</peticion>
      <!-- Nos indicará las paradas a realizar ordenadas y que paquetes se tiene
      que entregar o recoger en cada parada. -->
      <ruta>
            <parada>
                  <idPaquete>id</idPaquete>
                  <idPaquete>...</idPaquete>
            </parada>
            <parada>
                  ...
            </parada>
      </ruta>
</xml>
```